FACTORS AFFECTING ENERGY ABSORPTION OF A PLATE DURING SHOCK WAVE IMPACT USING A DAMAGE MATERIAL MODEL

By

Zachary Kyle Crosby

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Civil and Environmental Engineering
in the Department of Civil and Environmental Engineering

Mississippi State, Mississippi

August 2010

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
01 AUG 2010	**N/A**	**-**

4. TITLE AND SUBTITLE
Factors Affecting Energy Absorption of a Plate During Shock Wave Impact Using a Damage Material Model

5a. CONTRACT NUMBER
W56 HZV-08-C-0236 (SimBRS)

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Zachary Kyle Crosby

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Mississippi State University

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
US Army RDECOM-TARDEC 6501 E 11 Mile Rd Warren, MI 48397-5000, USA

10. SPONSOR/MONITOR'S ACRONYM(S)
TACOM/TARDEC

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
21017

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
A Thesis Submitted to the Faculty of Mississippi State University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Civil and Environmental Engineering in the Department of Civil and Environmental Engineering, The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**SAR**	**138**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

FACTORS AFFECTING ENERGY ABSORPTION OF A PLATE

DURING SHOCK WAVE IMPACT USING A

DAMAGE MATERIAL MODEL

By

Zachary Kyle Crosby

Approved:

Philip M. Gullett
Assistant Professor of Civil and
Environmental Engineering
(Major Professor)

Mark F. Horstemeyer
Professor of Mechanical Engineering
(Committee Member)

Thomas E. Lacy Jr.
Associate Professor of Aerospace
Engineering
(Committee Member)

James L. Martin
Professor of Civil and Environmental
Engineering
(Graduate Coordinator of the
Department of Civil and
Environmental Engineering)

Sarah A. Rajala
(Dean of Bagley College of Engineering)

Name: Zachary Kyle Crosby

Date of Degree: August 7, 2010

Institution: Mississippi State University

Major Field: Civil and Environmental Engineering

Major Professor: Dr. Philip Gullett

Title of Study: FACTORS AFFECTING ENERGY ABSORPTION OF A PLATE DURING SHOCK WAVE IMPACT USING A DAMAGE MATERIAL MODEL

Pages in Study: 124

Candidate for Degree of Master of Science

This thesis examines the influences of five factors on the strain energy at failure of metallic alloy plates during a shock wave impact. The five factors are material type, initial damage, boundary conditions, plate thickness, and plate temperature. The finite element simulation matrix was developed using a statistical design of experiments (DOE) technique. The Eulerian hydrocode CTH was used to develop the pressure histories that were input into the finite element code Abaqus/Explicit, which implemented the Mississippi State University internal state variable (ISV) plasticity-damage model (DMG). The DMG model is based on the Bammann-Chiesa-Johnson (BCJ) ISV plasticity formulation with the addition of porosity and the void nucleation, growth, and coalescence rate equations that admit heterogeneous microstructures. Material type and thickness were the primary influences on the strain energy at failure, and the materials studied, magnesium and aluminum, showed two different failure mechanisms, tearing at the boundaries and spalling, respectively.

ACKNOWLEDGEMENTS

First and above all, I would like to thank God the Father in the name of our Lord Jesus Christ for opening my eyes to the joy that comes from knowing Him. I thank Him for choosing me before the foundation of the world to know Him. Thanks also to Him for the opportunities that He opened up to me and for the gifts He has given to me. I did not choose the times or circumstances surrounding my life, but it is by His grace that I am where I am. *Soli Deo Gloria*.

Next, I would like to thank my advisor, Dr. Philip Gullett, who has guided me through my research and preparation of this thesis. I look forward to working with him on my PhD for the next three years, God-willing. I would also like to thank my committee members, Dr. Mark Horstemeyer and Dr. Thomas Lacy, who have given valuable advice and feedback on my research. My only regret is that I did not use their knowledge and expertise as fully as I could throughout the course of my Master's degree. I also greatly appreciate Dr. Richard Weed for his help with CTH. There was no one else around that could have helped me with my blast modeling, and I greatly appreciate his guidance in that area. Dr. Youssef Hammi helped me through some of my struggles with software and programming, and for that I am very thankful.

Without the support of my family, I would not have even been able to make it through my bachelor's degree, so I appreciate their support throughout all of my education. Thanks to my dad and mom, Cecil and Annice Crosby, for

their support, financially and through their encouragement. I also have great respect and thankfulness for my grandparents, Aubrey and Earline Allen. They have offered me tremendous support throughout my life and have taught me more than I could ever learn from any college course.

At last, I would like to thank my friends that I have spent the majority of the time with while working on this thesis. I appreciate Neil Williams for all of his help. I owe him a tremendous amount for his help in my research and for that time when I accidently knocked over a Dr. Pepper on a bunch of papers that were on his desk. Sorry, Neil. Matthew Priddy and Barrett Hardin have been with me as we each have gone through our first graduate school experience. I appreciate them for all that they have done and wish them both the best at Georgia Tech. Finally, thanks to TARDEC/DoD, CAVS, and the Bagley College of Engineering for their financial support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Blast mitigation has been the focus of increasing research efforts due to the growing threat of both civil and military blast attacks. The goal of this research effort is to incorporate a phenomenological material model into a simulation-based, blast-resistant design analysis, which should increase the understanding of the loading and failure mechanisms that are involved during an explosion. The phenomenological model used in this thesis is the DMG model, which consists of the Bammann-Chiesa-Johnson (BCJ) plasticity/damage model (Bammann and Aifantis, 1989) modified to include void nucleation, growth, and coalescence as damage (Horstemeyer and Gokhale, 1999). Using the DMG model with a simulation-based approach should improve the effectiveness of the design and lower the expenses and time spent in the blast-resistant design process by reducing the number of experiments needed by the traditional design process.

Blast mitigation is comprised of two aspects, ballistic protection and shock wave reduction. Ballistic protection requires the dissipation of kinetic energy from projectiles because projectiles cause damage when they come in contact with other bodies. Projectiles can consist of anything in the area of the explosion including the container around the explosive and soil particles in the case of underground explosions. Shock waves cause damage by propagating a nearly

instantaneous change in pressure and energy that causes deformation when contact is made with a solid object. Structures are damaged directly from shock waves due to high incident pressures. Human injuries can result from shock waves, directly and indirectly. Direct damage occurs because lungs, brains, and other organs are vulnerable to shock waves (Bellamy et al., 1991). Humans can also be injured indirectly by falling debris from damaged structures. The mechanisms involved in shock wave loading need further study so that sufficient protection from them can be created.

In structural responses, the failure mechanisms caused by blast-induced shock loads are as complicated as the blast loads themselves. The high-rate nature of blast loads require material characterization for high strain rate loading in order to better understand the structural response following a blast. This thesis analyzes the structural response of metallic alloy plate armor using a rate- and temperature-dependant internal state variable (ISV) material model, the DMG model, in a finite element analysis (FEA) in order to improve the understanding of the failure mechanisms involved in blasts and to improve design techniques for blast mitigation. The DMG model incorporates failure mechanisms by capturing the effects of the nucleation, growth, and coalescence of voids.

The simulation-based design technique applied in this thesis uses a design of experiments (DOE) approach made popular by Taguchi (Horstemeyer et al., 2003). The approach allows the study of the effects of many parameters while minimizing the number of experiments. The DOE approach was used to determine the influence of five factors on the energy absorbed in a plate from a

shock wave: material, plate thickness, temperature, initial damage, and boundary conditions. Conventional Weapons, or ConWep (2005), was originally used to determine the blast loads but did not accurately represent the blast loading needed in the FEA. To obtain accurate blast loads, the hydrocode CTH (2007) was used to generate the pressure histories for a given explosive and standoff distance. Those pressure histories were input into the finite element code Abaqus/Explicit (2009) to simulate the plate response. Determining the influences of the five factors on the plate response should improve blast mitigation design by focusing the design around parameters that will most influence the structural response.

The following chapter will describe the loading conditions from an explosion caused by shock waves. Then, an overview of the statistical DOE technique is given followed by a description of the material models used in the design simulations. The last two chapters present the setup of the shock wave and structural response models and explain the results from the simulation-based analysis, respectively.

CHAPTER II

SHOCK WAVE AND BALLISTIC LOADING THEORY

2.1 Introduction

The loading conditions on solid bodies in the vicinity of an explosion are complex due to high loading rates and complicated pressure and temperature variations. Blast loading is the result of two distinct sources resulting from an explosion: projectile impact and shock wave impingement. These two phenomena are the results of a sudden increase in volume of the explosive products and release of energy, usually from a chemical reaction in the explosive, which projects fragments away from the explosion along with a shock wave. Projectiles and shock waves can cause significant injuries and property damage to anything in the blast area; however, the loading conditions caused by the two events are entirely different, and the understanding of both should be furthered for better blast mitigation systems.

This chapter describes the loading conditions from an explosion caused by shock waves. The explosive process that produces these two phenomena will be described first. The loading conditions from shock wave impingement will be described next and will include shock front parameters, shock wave scaling laws, and shock wave loading on structures. A literature review on dynamic fragmentation will also be included. The information found in this chapter relies

heavily on Blast and Ballistic Loading of Structures by Smith and Hetherington (1994).

2.2 Generalities of Explosions

Explosions are caused by a rapid release in energy that can be classified into three unique categories: physical, nuclear, or chemical. The sudden failure of a pressure vessel, detonation of a hydrogen bomb, and detonation of a convention explosive such as TNT are examples of each category, respectively. Chemical explosives can be further categorized into high and low explosives, which are classified according to their burn rates. Low explosives deflagrate, or burn rapidly, while high explosives detonate, which is the process where a shock wave propagates through the explosive material initiating a chemical reaction behind it. Though some of the phenomena discussed here are applicable to all explosions, blasts from high explosives and shock waves in air are the focus of the information discussed in this thesis.

To initiate a blast from a high explosive, detonation of the explosive material must occur. As the detonation shock wave progresses through the explosive, the chemical reactions release large amounts of energy that generate hot gases at pressures from 10-30 gigapascals and temperatures from 3000 to 4000°C (Smith and Hetherington, 1994). The expansion of these gases forces the surrounding air from the space it was occupying, which creates a shock wave. The explosive gases move outward until their pressure and temperature reach equilibrium. The shock wave also moves outward until its pressures reach equilibrium. Further from the blast source, negative pressures can occur before

the shock wave reaches equilibrium. These negative pressures are caused by the inertia of the air that had been accelerated in the direction of the blast. This is not found very close to the explosive charge (Cooper, 1996).

2.3 Shock Waves and Blast Loading

2.3.1 Quantifying Shock Waves

A fixed point in air exposed to a shock front will experience significant pressure variations. An almost instantaneous rise to the maximum pressure occurs when the shock front contacts the fixed point. Any pressure above ambient pressure is known as overpressure. The pressures drop rapidly as the shock front passes until the pressures drop below ambient for a period of time. Figure 2.1 shows the pressure-time history for an ideal shock wave. The shape of the pressure-time curve depends on the standoff distance and type of explosive used.

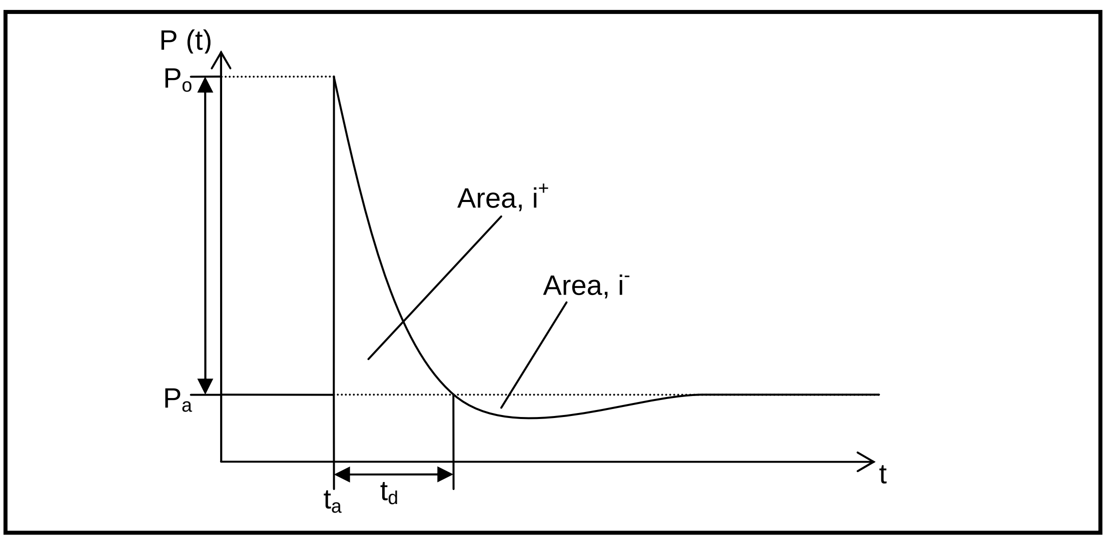

Figure 2.1 Pressure-time history of an ideal shock wave.

P_o is the peak overpressure, P_a is the ambient pressure, t_a is the arrival time, t_d is the duration of the positive overpressures or positive phase, i^+ is the impulse of the positive phase, i^- is the impulse of the negative phase.

2.3.2 Shock Front Parameters

Four types of pressure are considered in the study of shock waves: Incident overpressure, dynamic pressure, reflected pressure, and total or stagnation pressure. Incident overpressure, also known as hydrostatic pressure or side-on pressure, is the pressure found from a side-on measurement, or measurement tangential to the shock front. Incident overpressure is usually used to define the strength of a shock wave. Dynamic pressure is related to the kinetic energy of a fluid. Reflected pressure is associated with the reflection of a shock front from a surface. The reflected pressure is the effective loading for structures and can be many times greater than the measured incident pressures (Dharmasena, 2008). The total or stagnation pressure is the pressure found from a head-on measurement and is the sum of the incident overpressure and dynamic pressure with a compressibility factor (Braid, 2001).

The shock front parameters given by Smith and Hetherington (1994) are shock front velocity, air density behind the shock front, and maximum dynamic pressure and are given by

$$U_s = \sqrt{\frac{6p_s+7p_0}{7p_0}} \cdot a_0 \qquad \text{Equation 2.1}$$

$$\rho_s = \frac{6p_s+7p_0}{p_s+7p_0} \cdot \rho_0 \qquad \text{Equation 2.2}$$

$$q_s = \frac{5p_s^2}{2(p_s+7p_0)} \qquad \text{Equation 2.3}$$

where U_s is shock front velocity, ρ_s is the air density behind the shock front, q_s is the maximum dynamic pressure, p_s is the peak static overpressure, p_0 is the ambient air pressure ahead of the shock wave, ρ_0 is the density of the air at ambient pressure ahead of the shock wave, and a_0 is the speed of sound in air at ambient pressure.

The peak static overpressure from Brode (1954) according to Smith and Hetherington (1994) can be found according to

$$p_s = \frac{6.7}{Z^3} + 1\ bar\ \ (p_s > 10\ bar) \qquad \text{Equation 2.4}$$

$$p_s = \frac{0.975}{Z} + \frac{1.455}{Z^2} + \frac{5.85}{Z^3} - 0.019\ bar\ \ (0.1 < p_s < 10\ bar) \qquad \text{Equation 2.5}$$

where Z is the scaled distance given by

$$Z = \frac{R}{W^{\frac{1}{3}}}\,. \qquad \text{Equation 2.6}$$

In Equation 2.5, R is the distance from the center of the charge in meters and W is the charge mass in kilograms of TNT. TNT is universally used as the reference explosive for blast scaling. Blast scaling will be discussed more in the next section.

Henrych (1979) also described the peak static overpressure with equations similar to that of Brode (1954), as shown in Equations 2.7, 2.8, and 2.9. The accuracy of both Brode's and Henrych's solutions decreases as the distance from the center, z, decreases due to the formation of shock waves near the explosive.

$$p_s = \frac{14.072}{Z} + \frac{5.540}{Z^2} - \frac{0.357}{Z^3} + \frac{0.00625}{Z^4}\ bar\ \ (0.05 \le Z < 0.3) \qquad \text{Equation 2.7}$$

$$p_s = \frac{6.194}{Z} - \frac{0.326}{Z^2} + \frac{2.132}{Z^3}\ bar\ \ (0.3 \le Z \le 1) \qquad \text{Equation 2.8}$$

$$p_s = \frac{0.662}{Z} + \frac{4.05}{Z^2} + \frac{3.288}{Z^3}\ bar\ \ (1\ \leq\ Z\ \leq 10\ )$$ Equation 2.9

Another shock wave parameter is the area under the pressure-time curve known as the impulse, which can be used to describe a shock wave with a single value. The impulse of the positive phase is given by

$$i^+ = \int_{t_a}^{t_a+t_d} P(t)\, dt$$ Equation 2.10

Brode (1954) also had a solution for the maximum negative impulse during the negative phase given by

$$i^- \approx i^+\left(1 - \frac{1}{2\,Z}\right)$$ Equation 2.11

2.3.3 Shock Wave Scaling Laws

Blast scaling is used to predict the behavior of shock waves knowing the weight of the explosive and the distance from the explosion. Blast scaling is commonly used to scale incident overpressure values for a known charge mass and distance and is used in the program ConWep discussed in Chapter 5. The most common form of shock wave scaling is Hopkinson, or cube-root scaling (Baker, 1973). Hopkinson scaling implies that two charge masses W_1 and W_2 with diameters d_1 and d_2 of the same explosive material exhibit the relationship in Equation 2.12.

$$\frac{d_1}{d_2} = \left(\frac{W_1}{W_2}\right)^{\frac{1}{3}}$$ Equation 2.12

Assuming that $\frac{d_1}{d_2}$ = K, then Figure 2.2 shows the relationship between the shock wave parameters and the size of the explosive. The shock wave ranges,

R_1 and R_2, can also be calculated by an equation similar to Equation 2.12, shown as Equation 2.13 (Smith and Hetherington, 1994). It can be seen from Figure 2.2 that for a ratio of charge diameters $\frac{d_1}{d_2}$ = K, identical impulses can be produced by keeping the ratio of the ranges equal to K also.

$$\frac{R_1}{R_2} = \left(\frac{W_1}{W_2}\right)^{\frac{1}{3}} \qquad \text{Equation 2.13}$$

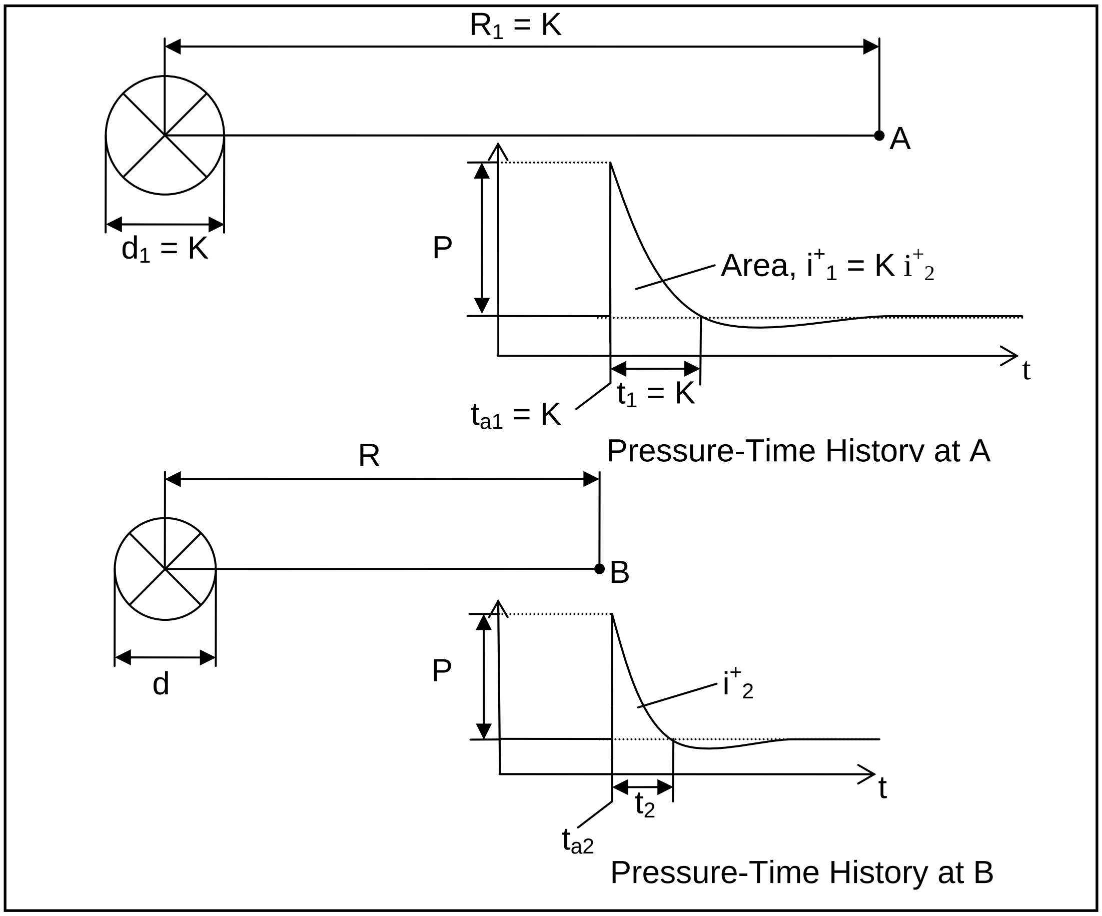

Figure 2.2 Hopkinson shock wave scaling (After Baker, 1973).

Another common scaling law is that of Sachs (1944) that takes into account the effects of high altitude where ambient conditions differ from those at sea level. However, these will not be discussed here.

2.3.4 Shock Wave Loading

Shock wave-induced structural loading, or simply shock wave loading, depends on a number of factors including size of the explosive, standoff distance, geometry of the target, and the angle at which the shock wave impacts the target. Variations in any one of those factors could result in loadings that differ by an order of magnitude. Shock wave loading for the simplest case, an infinitely large rigid wall with a zero degree angle of incidence, will be discussed first. Then, the effects of changing the angle of incidence and the structure size will be examined.

The simplest cases of shock wave loading occur on an infinitely large wall because no waves are able to diffract around the wall, which can greatly complicate the loading conditions. Loading for the wall with a zero degree angle of incidence is also known as the reflected overpressure. For this case, the reflected peak pressure can be found from the Rankine and Hugoniot (1870) equations derived in Equations 2.2 and 2.3 and is given by

$$p_r = 2p_s + \left(\frac{C_p}{C_v} + 1\right) q_s \qquad \text{Equation 2.13}$$

where C_p is the specific heat at constant pressure and C_v is the specific heat at constant volume. The dynamic pressure, q_s, is shown in Equation 2.14 and the velocity behind the shock front, u_s, is given in Equation 2.15.

$$q_s = \frac{1}{2} \rho_s \, u_s^2 \qquad \text{Equation 2.14}$$

$$u_s = \frac{a_0 p_s}{\frac{C_p}{C_v} p_0} \left(1 + \left[\frac{\frac{C_p}{C_v} + 1}{2 \frac{C_p}{C_v}} \right] \frac{p_s}{p_0} \right) \qquad \text{Equation 2.15}$$

where a_0, is the speed of sound. Smith and Hetherington (1994) show that Equation 2.13 can be shown as

$$p_r = 2 p_s \left[\frac{7 p_0 + 4 p_s}{7 p_0 + p_s} \right] \qquad \text{Equation 2.16}$$

which indicates that an upper and lower limit on the reflected pressure can be set. The lower limit occurs when the incident overpressure is a lot less than the ambient pressure, and the upper limit occurs when the incident overpressure is much greater than the ambient pressure. These two cases result in the simplification of Equation 2.16 into the upper and lower limits shown in Equations 2.17 and 2.8.

$$p_r = 2 p_s \qquad \text{Equation 2.17}$$

$$p_r = 8 p_s \qquad \text{Equation 2.18}$$

The ratio between the reflected peak pressure and incident overpressure is known as the reflection coefficient, C_R. From the Rankine and Hugoniot (1870) equations, the reflection coefficient is found in the range $C_R = 2 - 8$ (Smith and Hetherington, 1994). However, these equations assume ideal gas conditions that may not be valid for a large blast that generates a large temperature and pressure jump. The reflection coefficient for those conditions can be as large as twenty (Wilkinson and Anderson, 2003).

The angle of incidence, α_i, of the shock wave on the surface of the structure also plays a role in determining the reflection coefficient. In the previous discussion where $\alpha_i = 0°$, the reflection coefficient could be between $C_R = 2 - 8$ for ideal gas conditions or as high as $C_R = 20$ without ideal gas conditions. When $\alpha_i = 90°$, there is no reflection and the loading is simply the peak overpressure. Between these limits, two different types of reflection can occur that affect the reflection coefficient: regular reflection and Mach reflection (Smith and Hetherington, 1994). The effects of these types of reflection are not discussed here, but Figure 2.3 shows the effects of α_i on the reflection coefficient for various peak positive incident overpressures.

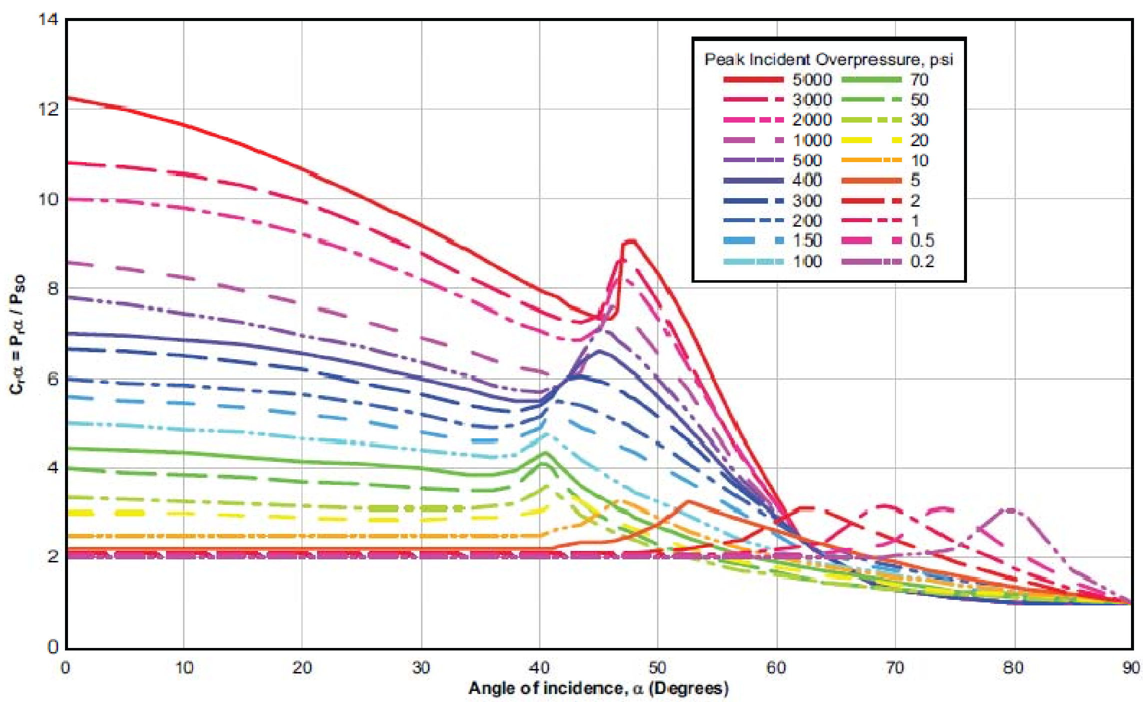

Figure 2.3 Reflection coefficient vs. angle of incidence for varying values of incident overpressure (from *Structures to Resist the Effects of Accidental Explosions*, 1990).

The last factor affecting the magnitude of the shock wave loading considered here is the geometry of the target. In the case of the infinite wall, no diffraction can occur, which greatly simplifies the loading conditions and allows us to set the limits for reflected pressure. However, those loading conditions are not typical, and the geometry of the target must be taken into consideration to represent realistic conditions surrounding a blast. A rigid, rectangular structure is discussed here to examine the shock wave interaction with a target of simple geometry as shown in Figure 2.4.

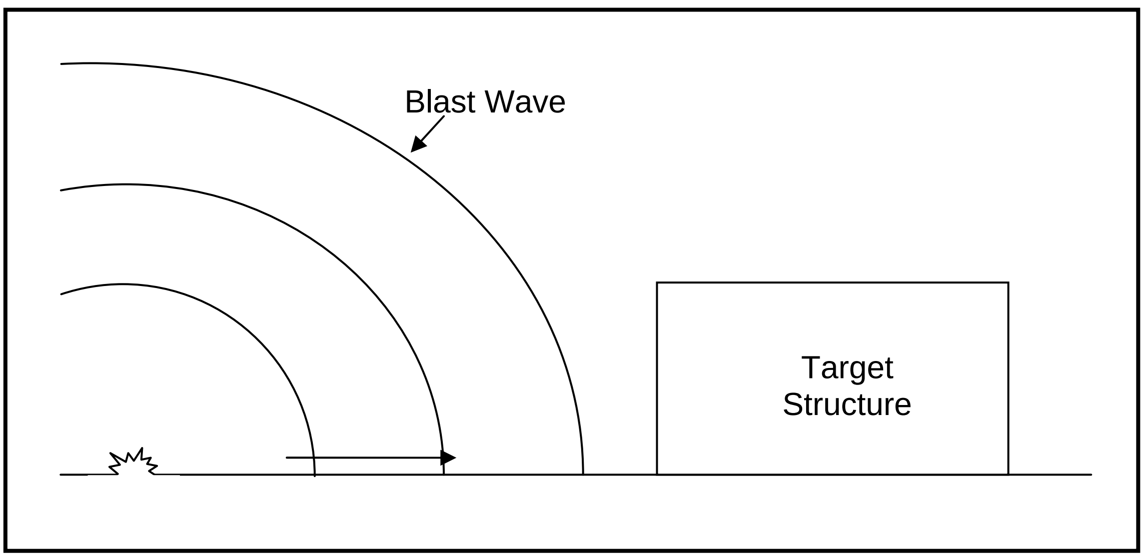

Figure 2.4 Shock wave approaching target structure.

When the front face of the structure is impacted by the shock wave, a reflected wave is formed and the face is loaded with the reflected pressure, p_r. As the reflected wave reaches the free edges of the front face, the air flows from the high pressure region to the low pressure region around the edges of the front face. This flow of air moves from the free edges and progresses towards the

center of the front face as a rarefaction or relief wave. As the relief wave arrives at the center, it decreases the reflected pressure. The reflected pressure decreases until the pressure is equal to the stagnation pressure p_{stag} given by

$$p_{stag} = p_s + C_D\, q_s \qquad \text{Equation 2.19}$$

where p_s is the peak top and side overpressure, C_D is the drag coefficient for the front face, and q_s is the dynamic pressure (Rickman and Murrell, 2007). Once the shock wave diffracts around the free edges, the overpressure will load the top and sides of the structure. There will also be a drag loading on the structure that will push the structure in the direction that the shock wave is traveling. The drag force is given by

$$F_D = C_D\, q_s\, A \qquad \text{Equation 2.20}$$

where A is the area loaded by the pressure. Once the shock front has passed, the structure is loaded with a suction force caused by the dynamic pressure passing over and around the structure. A more detailed description of shock wave external loading on structures can be found in Smith and Hetherington (1994).

2.4 Ballistic Loading

2.4.1 Literature Review on Dynamic Fragmentation

Dynamic fragmentation is the separation of a body of material into discrete sections caused by an impulsive loading. Many materials fail by dynamic fragmentation when subjected to a rapid increase in energy. Sources of impulse loading may include an impact with another body, rapid temperature change, or

impingement of a shock wave. The fundamental mechanisms of dynamic fragmentation have received little attention, though the prevention or controlling of this phenomenon is of importance especially in defense applications. Furthermore, brittle materials have been the focus of most research with regard to dynamic fragmentation and will be the only materials discussed here.

Brittle materials fail under slow strain rates by the growth of a single dominant crack across the entire section of a body. The crack is usually nucleated from an initial flaw in the body due to a material defect. Because of the slow loading, this single crack relieves the stresses at other potential crack nucleation sites allowing only the single crack. When high strain rates are applied, large stresses occur in a relatively short amount of time allowing for increased crack nucleation at many sites. Without enough time to relieve the stresses, many cracks nucleate and propagate simultaneously through the material that coalesce to allow the material to separate into fragments. This describes the phenomenon of dynamic fragmentation in brittle materials.

Many models have been developed to describe dynamic fragmentation with the main goal of predicting fragment size distributions. The earliest models use energy balance principles to predict fragment sizes. Grady (1982), Glenn and Chudnovsky (1986), Yatom and Ruppin (1989), and Yew and Taylor (1994) are some of the theoretical models used that apply energy balance principles. Other models use statistical methods to determine fragment size distributions such as Grady and Kipp (1985), Grady (1990), and Zhou et al. (2006). More recent models have emphasized the importance of determining time to fragmentation onset in addition to fragment sizes. These models include the

analytical models of Drugan (2001) and Miller et al. (1999) and the computational models of Camacho and Ortiz (1996), Espinosa et al. (1998), Miller et al. (1999), and Cirak et al. (2005).

The energy balance models began with a simple balance of local kinetic energy and surface energy and became increasingly more complex with the addition of stored elastic energy and thermodynamic principles. When computational models became popular, the energy balance models were shown to be applicable only at very high strain rates (greater than 10^7 s^{-1}) because they assume that the energy available to form fracture surfaces causes the surfaces to form instantaneously. Recent studies have shown that the time history of the fragmentation process and the speed of crack propagation are important for a model to be valid at all testing regimes.

The computational and advanced analytical models recently developed include the effects of time on dynamic fragmentation and are able to predict the fragment size distribution more accurately than the energy balance models. The majority of computational models available are finite element models based on a cohesive law. A cohesive-law model forms a body of material by joining prospective fragments with nonlinear cohesive zones to the rest of the body. This allows the prediction of fragment sizes with the consequent inclusion of time to fragmentation initiation as a function of material properties and the applied strain rate. This model is more powerful than the energy balance models in that it can account for discrete cracks where the energy balance models cannot. The analytical models from Drugan (2001) and Miller et al. (1999) are also based on a cohesive law.

Though the computational and analytical models mentioned previously are far more comprehensive than the foundational energy balance models, several areas are still in need of research before they can be adequately understood. Experimentation is needed to validate the current models across a wider regime of problems that may be faced in applications involving dynamic fragmentation. Material characterization is also needed in regard to this phenomenon to gain insight as to how different materials behave during dynamic fragmentation. Moreover, the current models should be tested for their validity with ductile materials. If the models cannot adequately determine the behavior of ductile materials in dynamic fragmentation, new models should be researched that have this ability.

CHAPTER III

DESIGN OF EXPERIMENTS

3.1 Introduction

The design of experiments (DOE) methodology is comprised of statistical techniques developed by Ronald Fisher (1935, 1936) to relate statistical procedures to physical experiments. Fisher's studies focused on reducing the amount of experiments needed when the effects of a large amount of factors were being studied. As the number of experiments became too large due to the presence of many factors, he created methods to reduce the total number of combinations needed such that all of the factors would be evenly present while also capturing the effects of the factors. His techniques demonstrated how to study the effects of multiple variables simultaneously. As the theory behind his techniques became more complicated, fewer people chose to use them in practical applications. Genechi Taguchi alleviated this problem in the field of quality engineering by developing a modified and standardized form of DOE which has been utilized in this paper (Roy, 2001).

The Taguchi approach has been used in various contexts of mechanics and design by Trinh and Gruda (1991), Horstemeyer (1993), Stutsman et. al (1996), Young (1996), Nedler and Lee (1991), Horstemeyer and McDowell (1997), Horstemeyer et. al (1999, 2003), and Horstemeyer and Ramaswamy (2000). This approach allows the evaluation of multiple parameters in an efficient

manner (Horstemeyer et. al, 2003). In the context of this paper, the DOE technique is used to determine the influences of five parameters on the energy absorption from a shock wave. These parameters are placed in an orthogonal array, standardized by Taguchi, which allows the optimal determination of parametric effects. The parameters used in the orthogonal array are discussed in the next section followed by an explanation of the DOE methodology.

3.2 Parameters

Five parameters (material, damage distribution, boundary conditions, plate thickness, and temperature) are examined in this analysis to determine their effect on the energy absorption in a metallic alloy plate. These parameters were chosen so that they would represent conditions around a driver-side floor pan of an Armed Forces Hummer.

3.2.1 Material

Four materials have been chosen to determine how each would respond to the shock wave loading: cast AM60 magnesium alloy, rolled AZ31 magnesium alloy, cast A356-T6 aluminum alloy, and extruded 6061-T6 aluminum alloy. The AZ31 magnesium alloy was chosen to determine its blast mitigation properties relative to other materials that have been characterized for the DMG model, discussed in Chapter 4, and based off of military specifications for AZ31 magnesium alloy (‘*Detail Specification Armor Plate, Magnesium, AZ31B, Applique’*, 2009). The 6061 aluminum alloy was chosen because it is also used in plate armor applications, and the AM60 magnesium alloy and A356 aluminum alloy were chosen solely for comparison with the other metallic alloys.

3.2.2 Damage Distribution

The DMG model predicts failure using a damage ISV approach where void nucleation, growth, and coalescence are included as damage. As such variation of initial damage is considered, four damage distributions are used to determine how varying damage affects the amount of energy absorbed by the plate. A homogeneous distribution is used and should be able to absorb the most energy. Three heterogeneous distributions are also used that have the same mean damage, but the standard deviation of the damage is varied. The mean initial damage is based off of the initial void volume fraction for each material that can be found in the appendix. The three heterogeneous distributions have 30 percent, 60 percent, and 100 percent standard deviation from the mean and each distribution should have effects that increase as the standard deviation increases.

3.2.3 Boundary Conditions

Boundary conditions in a numerical model play a significant role in determining the response of the model. As discussed in Chapter 5, a 3D finite element model was used to calculate the energy absorption of a plate. Pinned-like and clamped boundary conditions will be used in the structural response model to see how the results are affected. The pinned-like and clamped conditions should bound the minimum and maximum response, respectively, with realistic conditions yielding a response between the two. The force boundary conditions will be the same for all of the simulations, which are the pressures applied from the Abaqus/Explicit VDLOAD subroutine. The entire model

including the boundary conditions and subroutine will be discussed in detail in Chapter 5.

3.2.4 Plate Thickness

Plate thickness is expected to be the dominant influence in the response of the plate. Four plate thicknesses were chosen, 10, 30, 50 and 75 mm. The minimum and maximum plate thickness values were chosen based on *Detail Specification Armor Plate, Magnesium, AZ31B, Applique* (2010). The middle values were chosen somewhat arbitrarily, but they can be used to further optimize the plate thickness for a given shock wave if desired.

3.2.5 Initial Plate Temperature

Since plasticity is temperature dependant, temperature is expected to have some influence on the response of the plate following the shock wave impact. The temperatures used in this thesis were used to represent the normal operational conditions for a vehicle. However, a small temperature range (273 K to 323 K) is considered so its effects are expected to be limited. Also, the AM60 is the only material for which mechanical response at various temperatures was available.

3.3 Design of Experiments Methodology

To evaluate the influences of the five parameters, 16 finite elements calculations are set up in an orthogonal array according to the Taguchi method. An orthogonal array is generated to govern the arrangement of the five parameters in the 16 calculations. After the calculations are completed according to their arrangement in the array, the DOE method is used to

determine the influences of the parameters on the response, energy absorption of the plate. The value of this approach is easily seen when compared to another DOE method, the full factorial approach. The full factorial approach examines every combination of parameters resulting in $2(4^4)$, or 512, simulations to determine the parametric influences as compared to the 16 calculations performed in this thesis. The concept of the orthogonal array is discussed next followed by the procedure for determining the parametric influences. A more detailed description can be found in Box et al. (1978).

The orthogonal array used in this thesis is the mixed $L_{16}(4^4\ 2^3)$ array where 16 is the number of experiments (finite element calculations), 4 and 2 are the number of levels for the respective parameters, and 4 and 3 are the number of parameters that can be investigated. However, only 2^1 is used in this thesis for the single two level parameter, boundary condition; the other two orthogonal array columns for extra two level parameters are zeroed out. The number of levels refers to the different options for each parameter, e.g., AM60, AZ31, A356, and 6061 are the four levels under the parameter material as seen in Table 3.1. In a typical array, each level has an equal number of occurrences in each column. Since the columns are orthogonal, or independent of each other, large variations in the results when changing levels of a factor show that the factor has a strong influence on the design response parameter (energy absorption for this analysis). The effects of the other factors are canceled out because the levels of the other factors occur an equal number of times (Peace, 1993 cited in Young, 1996). The L_{16} array can be seen in Table 3.1. After the parameters have been

determined and the array has been created, the calculations can be run and the parametric influences can be found.

Table 3.1 L_{16} orthogonal array showing the parameters.

Calculation	Material	Thickness	Temperature	Damage Distribution	Boundary Conditions
1	AM60 Magnesium	10 mm	323 K	100% St. Dev.	Clamped
2	AM60 Magnesium	30 mm	303 K	60% St. Dev.	Clamped
3	AM60 Magnesium	50 mm	293 K	30% St. Dev.	Pinned-like
4	AM60 Magnesium	75 mm	273 K	0% St. Dev.	Pinned-like
5	AZ31 Magnesium	10 mm	303 K	30% St. Dev.	Pinned-like
6	AZ31 Magnesium	30 mm	323 K	0% St. Dev.	Pinned-like
7	AZ31 Magnesium	50 mm	273 K	100% St. Dev.	Clamped
8	AZ31 Magnesium	75 mm	293 K	60% St. Dev.	Clamped
9	A356 Aluminum	10 mm	293 K	0% St. Dev.	Clamped
10	A356 Aluminum	30 mm	273 K	30% St. Dev.	Clamped
11	A356 Aluminum	50 mm	323 K	60% St. Dev.	Pinned-like
12	A356 Aluminum	75 mm	303 K	100% St. Dev.	Pinned-like
13	6061 Aluminum	10 mm	273 K	60% St. Dev.	Pinned-like
14	6061 Aluminum	30 mm	293 K	100% St. Dev.	Pinned-like
15	6061 Aluminum	50 mm	303 K	0% St. Dev.	Clamped
16	6061 Aluminum	75 mm	323 K	30% St. Dev.	Clamped

After determining the response (energy absorption) of each calculation, Minitab-15 statistical software (2007) was used to determine how each parameter affected the response. This software allows the input of the parameters and responses (energy absorption for each calculation) and determines the parameters that most influence the response. To determine the influences of the parameters, the mean response for each level of the parameters is determined (e.g., the average energy absorption for the level AZ31 Mg alloy for the parameter material). Then, the difference between the highest and lowest average value for each level of a parameter measures the amount of influence a parameter has on the response. For example, in this analysis the

parameter boundary condition had the smallest difference between the average energy absorption for the calculations using clamped (level 1) boundary conditions and the average energy absorption for the calculations using the pinned-like (level 2) boundary conditions. Therefore, the parameter boundary condition had the smallest influence on the response (energy absorption). The results for the other parameters can be found in Chapter 6.

The DOE analysis can be used to determine the parametric influences for any set of responses. For this thesis, energy absorption by the plate is the response of interest. Other responses could easily be studied such as the amount of damage created or maximum deflection. However, a separate statistical analysis must be run for each set of responses. Once the parametric influences are determined for the responses, the parameters can be adjusted to achieve an optimal design. In this case, the optimal design would consist of the best plate design and conditions to increase energy absorption during a shock wave impact.

CHAPTER IV

MATERIAL MODELS

4.1 Introduction

Several material models were used in the hydrocode CTH and finite element code Abaqus/Explicit. In CTH, three material models were used for the air, soil, and TNT. The air model used the equations of state (EOS) based on the work of Graboske (1975), the soil model used the CTH geological strength model using data from Kerley (2001), and the TNT model used the Jones-Wilkins-Lee (JWL) EOS (Lee, 1968). The DMG model, or modified BCJ plasticity-damage model, was used to model the structure in Abaqus/Explicit. These models are discussed in detail in the following sections.

4.2 CTH

CTH is a shock physics code used to predict the pressure histories of a given explosive. Three material models were used from the CTH material model database called SESAME that contained tabular EOS for air, the CTH geological soil model was used for the soil, and the JWL EOS was used for TNT. These models are shown in this section.

4.2.1 Air

The default SESAME tabular EOS were used for air. An EOS is a relationship between the volume, pressure, quantity, and temperature of a

material in a given state (Cooper, 1996). The SESAME option in CTH uses EOS tables in the CTH database that are constructed using EOS modeling codes. The tables are intended to cover a broad density-temperature range and can include tables based on the Mie-Gruneisen model, the Debye specific heat, and other extensions to treat high compressions, large expansions, and high temperatures. Some tables may even include theories that treat chemical and physical phenomena (Hertel Jr. and Kerley, 1998). The SESAME tabular equations for air are based on Graboske (1975).

4.2.2 Soil

Sand was the soil used in all of the CTH simulations. Sand is a porous aggregate of granular materials commonly composed of crushed rock and mineral particles such as silica. Sand properties vary depending on the degree of compaction, the minerals present, and the moisture content (Kerley, 2001). Dry sand was used in the initial simulations, but wet sand was incorporated in the latter ones because it was shown by Kerley that an explosive in wet sand produces a larger impulse than the same explosion in dry sand.

The initial CTH simulations that incorporated soil models used dry sand with a zero initial air void porosity and zero percent water content. This should give unrealistic pressure histories and was not used for the structural response simulations. The reference state properties for dry sand are shown in Table 4.2.

The CTH simulations that produced the pressure histories used in the structural response simulations used wet sand in the soil model. Kerley (2001) shows that dry sand is able to absorb more explosive energy than wet sand

because the energy absorption of dry sand is 50 to 100 percent greater than for wet sand at the same shock pressure. The wet sand soil model is described in Kerley (2005).

The wet sand properties are based on quartz sand with a 24% water content. The grain density of the quartz is 2.648 g/cm3. With no air filled voids, the density of the sand with the 24% water is 2.0028 g/cm3. The strength of the soil is found using the CTH geological strength model given in Equation 4.1 in Kerley (2005) from McGlaun et al. (1988).

$$Y(P) = Y_0 + \Delta Y(1 - e^{\frac{-Y_P P}{\Delta Y}}) \qquad \text{Equation 4.1}$$

where $\Delta Y = Y_{max} - Y_0$, Y_0 is the zero-pressure strength, Y_p is the pressure derivative, and Y_{max} is the maximum strength. The parameters used for wet sand in CTH are shown in Table 4.1.

Table 4.1 Properties for dry and wet sand used in CTH.

Soil Properties	
Dry Sand	
Initial Density (g/cc)	2.605
Fracture Strength (dynes/cm2)	-1.0E+6
Sound Speed (km/s)	3.883
Wet Sand	
Initial Density (g/cc)	1.97
Fracture Strength (dynes/cm2)	-1.0E+6
Sound Speed (km/s)	1.5499
Crush Pressure (dynes/cm2)	5.0E+8
Yield Strength (dynes/cm2)	2.0E+8
Zero-Pressure Strength (dynes/cm2)	3.0E+6
Pressure Derivative	2
Poisson Ratio	0.32
Melt Temperature (electron volts)	0.15

4.2.3 TNT Explosive

The explosive was modeled using the default JWL EOS for TNT. The JWL formulas were developed to represent the EOS for explosive detonation products. The TNT in the CTH simulation was detonated by programmed burn (HEBURN) at the bottom center of the explosive with a detonation velocity of 6.93E5 cm/s (Dobratz and Crawford, 1985, cited in Kerley, 2001). Though the JWL EOS have been shown to predict lower pressures as standoff distance increases (Kerley, 2005), it is commonly used and gives an approximate pressure history given the large number of variables involved.

4.3 DMG Damage Model

The finite element analyses performed in this paper used the DMG model (Horstemeyer et al., 2000). The DMG model is an ISV plasticity/damage model based on the microstructure-property model equation of Bammann and Aifantis (1989) and Horstemeyer and Gokhale (1999). The DMG model predicts damage evolution by incorporating separate void nucleation, growth, and coalescence evolution equations in the BCJ ISV plasticity model. The ISVs in this model are used to describe the macroscale effects of microstructural features (e.g., voids, cracks, and inclusions). These microstructural features are dependant on the material, forming process, environment in which it is used, and the loading conditions present. Using this concept, the present state of the material depends only on the present values of the observable variables and the ISVs, which allows the model to implicitly capture the loading history of the material.

The DMG model can be divided into three major aspects: kinematics; void nucleation, growth, and coalescence; and plasticity. The kinematics in the

continuum damage mechanics framework are described first. Then, the void nucleation, growth, and coalescence aspects of the model are discussed. Lastly, the elastic-plastic aspects of the macroscale model are discussed.

4.3.1 Kinematics

The formulation of the kinematics follows that of Davison et al. (1977) and Bammann and Aifantis (1989) as described in Horstemeyer et al. (2000). The kinematics of motion are described by elastic straining, inelastic flow, and formulation and growth of damage. As shown in Figure 4.1, the deformation gradient, $\underline{F}$, is decomposed into the plastic, ($\underline{F}_d^p$), dilatational inelastic ($\underline{F}_v^p$), and elastic parts ($\underline{F}^e$) given by

$$F = \underline{F}^e \underline{F}_v^p \underline{F}_d^p. \qquad \text{Equation 4.2}$$

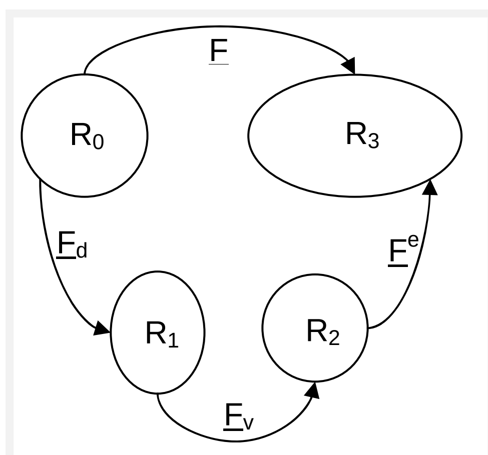

Figure 4.1 Multiplicative decomposition of the deformation gradient (After Horstemeyer et al., 2007).

For crystalline metallic alloys, $\underline{F}^e$ represents lattice displacements from equilibrium, $\underline{F}_d^p$ represents a continuous distribution of dislocations whose motion produces permanent shape changes but no volume change, and $\underline{F}_v^p$ represents a

continuous distribution of voids contributing to permanent volume change. $\underline{F}_v^p$ is the deformation gradient that describes the damage and is discussed in detail in the following paragraphs.

Porosity, or volume of voids per unit total volume, is used as the measure of damage because it allows the evolution of damage to be expressed directly in terms of nucleation and growth rates of the void population (Bammann and Aifantis, 1989). From the multiplicative decomposition used here, damage, Φ, can be written as

$$\Phi = \frac{V_V}{V_2}, \qquad \text{Equation 4.3}$$

where V_V is the volume of voids and V_2 is the volume of an element in the elastically unloaded state, R_2. Equation 4.3 defines damage as the ratio of the change in volume of an element in the elastically unloaded state, R_2, from its volume in the reference state, R_0. The change in volume from R_2 to R_0 is the volume of voids, V_v, is the only volume change to occur between the different states; there is no volume change between R_0 and R_1 due to inelastic incompressibility. From Equation 4.3, it follows that

$$V_0 = (1 + \Phi)V_2 \ , \qquad \text{Equation 4.4}$$

The Jacobian of the deformation gradient is related to the change in volume or change in density for constant mass as

$$J = \det \underline{F}_v^p = \frac{V_2}{V_0} = \frac{\rho_0}{\rho_2} \qquad \text{Equation 4.5}$$

and must be positive. Combining Equation 4.4 and 4.5, yields

$$J = \det \underline{F}_v^p = \frac{1}{1 - \Phi} \ . \qquad \text{Equation 4.6}$$

where the Jacobian is now determined by the damage parameter, Φ. For simplicity, the damage is assumed to produce isotropic dilation, which gives the volumetric part of the deformation gradient as

$$\underline{F}_v^p = \frac{1}{(1-\Phi)^{1/3}} \underline{I} \ , \qquad \text{Equation 4.7}$$

where $\underline{I}$ is the second-order identity tensor The velocity gradient associated with the deformation gradient, $\underline{L} = \dot{\underline{F}}\, \underline{F}^{-1}$, from Equation 4.2 is given by

$$\underline{L} = \underline{L}^e + \underline{L}_v^p + \underline{L}_d^p \qquad \text{Equation 4.8}$$

Similar formulas exist for the elastic, volumetric plastic, and deviatoric plastic parts of the velocity gradients as $\underline{L} = \dot{\underline{F}}\, \underline{F}^{-1}$. The volumetric part of the deformation gradient reads

$$\underline{L}_v^p \equiv \dot{\underline{F}}_v^p\, \underline{F}_v^{p^{-1}} = \frac{\dot{\Phi}}{3\,(1-\Phi)} \underline{I} \ . \qquad \text{Equation 4.9}$$

This defines the plastic volumetric rate of deformation as

$$\underline{D}_v^p = \frac{\dot{\Phi}}{3\,(1-\Phi)} \underline{I}. \qquad \text{Equation 4.10}$$

The trace of the plastic volumetric rate of deformation shows that the damage parameter, Φ, directly relates to the volumetric rate of deformation, as seen in Equation 4.11.

$$tr\left(\underline{D}_v^p\right) = \frac{\Phi}{(1-\Phi)}, \qquad \text{Equation 4.11}$$

The symmetric parts of the velocity gradient decompose similar to Equation 4.8.

$$\underline{D} = \underline{D}^e + \underline{D}_v^p + \underline{D}_d^p \ , \qquad \text{Equation 4.12}$$

When the rate of deformation related to the damage is defined, the damage in terms of void nucleation and growth can be described. The total number of voids in the reference configuration, R_0, is assumed to be N in the representative continuum volume, V_0. Let η^* represent the number of voids per unit volume in the reference configuration so that $\eta^*=N/V_0$. The average void volume becomes $v_v = N^{-1} \sum_{i=1}^{N} v_i$. The volume of voids is

$$V_v = \eta^* V_0 v_v. \qquad \text{Equation 4.13}$$

Equation 4.3 can then be written as

$$\Phi = \frac{\eta^* V_0 v_v}{V_0 + \eta^* V_0 v_v} = \frac{\eta^* v_v}{1+\eta^* v_v}. \qquad \text{Equation 4.14}$$

If the number of voids per unit volume, η, is defined in the intermediate configuration, then

$$\Phi = \frac{V_V}{V_2} = \frac{V_V}{N}\frac{N}{V_2} = v_v \eta, \qquad \text{Equation 4.15}$$

where

$$\eta = \frac{N}{V_2} = \frac{N}{V_0}\frac{V_0}{V_2} = \eta^* \frac{V_0}{V_2}. \qquad \text{Equation 4.16}$$

Using $V_2 = V_0 - V_v$ and Equation 4.3, the previous equation becomes

$$\eta^* = \frac{\eta}{(1-\Phi)}. \qquad \text{Equation 4.17}$$

4.3.2 Void Nucleation, Growth, and Coalescence

Three different phenomena related to damage initiation and progression are accounted for in the DMG model: void nucleation, growth, and coalescence. These three phenomena and how they are accounted for in the model will be

discussed beginning with void nucleation and followed by void growth and coalescence.

A modified version of the void nucleation rule of Horstemeyer and Gokhale (1999) is used in this model. This version includes a multiplier of the effective stress to capture strain rate or other stress variation effects on the nucleation rate (Tucker et al., 2010). The void nucleation rate equation is given by

$$\eta(t) = C_{coeff} \exp\left\{\frac{\varepsilon(t) d^{\frac{1}{2}}}{K_{IC} f^{\frac{1}{3}}} \frac{V(T)}{Y(T)} J_2 \left[a \left(\frac{4}{27} - \frac{J_3^2}{J_2^3} \right) + b \frac{J_3}{J_2^{\frac{3}{2}}} + c \left\| \frac{I_1}{\sqrt{J_2}} \right\| \right] \right\} \exp\left({}^{-\mathrm{C}_{\eta\mathrm{T}}}/_{\mathrm{T}} \right) \quad \text{Equation 4.18}$$

where $\eta(t)$ is the void nucleation density, $\varepsilon(t)$ is the strain magnitude at time *t*, C_{coeff} is a material constant, and T is the temperature. The material parameters *a*, *b*, and *c* relate to the volume fraction of the nucleation events caused by local stresses in the material. The void nucleation rate equation is stress state dependent due to the inclusion of the stress invariants I_1, J_2, and J_3. I_1 is the first invariant of stress, J_2 is the second invariant of deviatoric stress, and J_3 is the third invariant of deviatoric stress. The volume fraction of the second phase material is *f*, the average silicon particle size is *d*, and the bulk fracture toughness is K_{IC}. $\mathrm{C}_{\eta\mathrm{T}}$ is the void nucleation temperature dependent parameter. V(T) determines the magnitude of rate-dependence on yielding and Y(T) is the rate-independent yield stress. The motivation for the choice of stress invariants can be found in Horstemeyer and Gokhale (1999).

Void growth is the next phenomena to consider when determining the damage state. The damage framework described in the previous section allows

for different growth rules; however, the growth rule currently being used is that of McClintock (1968),

$$v_V = \frac{4}{3}\left\{R_0\, exp\left[\varepsilon(t)\ \frac{\sqrt{3}}{2(1-n)} \times\ sinh\left(\sqrt{3}(1-n)\,\frac{\sqrt{2}I_1}{3\sqrt{J_2}}\right)\right]\right\}^3 \qquad \text{Equation 4.19}$$

The material constant *n* is related to the strain hardening exponent. R_0 is the initial radius of the voids. From Equation 4.19, the void volume increases as the strain and/or stress triaxiality increases.

The last phenomena related to damage taken into account in the DMG model is void coalescence, the joining of voids. Coalescence has been observed to occur by two mechanisms, natural coalescence (Garber et al., 1976) and by a void sheet mechanism (Cox and Low Jr., 1974 and Rogers, 1960). Natural coalescence occurs when two neighboring voids grow together and become one. The void sheet mechanism is the process when a localized shear band occurs between neighboring voids. As described in Horstemeyer (2001), natural coalescence and the void sheet mechanism of coalescence arises naturally with the multiplicative relation between the nucleation and growth terms as

$$C = \ (C_{D1} + C_{D2}\,\eta\, v)\left(\frac{DCS_0}{DCS}\right)^z exp\ \left({}^{-C_{CT}}\!/_{T}\right) \qquad \text{Equation 4.20}$$

The parameters C_{D1} and C_{D2} are material constants that affect natural coalesce and the void sheet mechanism, respectively. C_{CT} is the void coalescence temperature dependent parameter. DCS_0, DCS, and z are parameters to capture the microstructure effect of grain size. The dependence on dendrite cell size (DCS) is based on the work of Major et al. (1994).

4.3.3 Plasticity

The BCJ ISV plasticity model (Bammann et al., 1993) is used to model the elastic-plastic aspects of the DMG model. The void nucleation, growth, and coalescence aspects mentioned in the previous section are inserted into the BCJ ISV model to account for stress state dependent damage evolution. The relevant equations in this model are indicated by the rate of change of the observable variables and internal state variables. The equations used within the finite element method framework include a linear, isotropic material response

$$\overset{0}{\underline{\sigma}} = \lambda\,(1-\Phi)\,tr\left(\underline{D}^{e}\right)\underline{I} + 2\mu(1-\Phi)\underline{D}^{e} - \left(\frac{\dot{\Phi}}{1-\Phi}\right)\underline{\sigma}\ , \qquad \text{Equation 4.21}$$

Equation 4.22 shows the additive decomposition of the strain rate followed by Equation 4.23 showing the rate and temperature-dependent evolution equation for plastic flow.

$$\underline{D}^{e} = \underline{D} - \underline{D}^{p}\ , \qquad \text{Equation 4.22}$$

$$\underline{D}^{p} = f(T)\sinh\left\{\frac{\|\underline{\sigma}\prime - \underline{\alpha}\| - [R+Y(T)]\,[1-\Phi]}{V(T)\{1-\Phi\}}\right\}\frac{\underline{\sigma}\prime - \underline{\alpha}}{\|\underline{\sigma}\prime - \underline{\alpha}\|}\ , \qquad \text{Equation 4.23}$$

where $\underline{\sigma}$ and $\overset{0}{\underline{\sigma}}$ are the Cauchy stress and the co-rotational rate of the Cauchy stress, respectively. As mentioned earlier, Φ is an ISV representing damage with $\dot{\Phi}$ representing its material time derivative. λ and μ are the elastic Lame constants, and $\underline{D}^{e}$ is the elastic deformation tensor. Equation 4.23 shows the plastic deformation tensor or inelastic flow rule, $\underline{D}^{p}$, where $\underline{\sigma}'$ is the deviatoric part of the stress tensor, T is temperature, $\underline{\alpha}$ is kinematic hardening, and R is isotropic hardening.

Equations 4.24 to 4.26 are functions related to yielding with Arrhenius-type temperature dependence and modified to account for stress state dependence.

$$Y(T) = C_3 \left[1 + C_a \left(\frac{4}{27} - \frac{J_3^2}{J_2^3}\right) + C_b \frac{J_3}{J_2^{\frac{3}{2}}}\right] e^{\left(\frac{C_4}{T}\right)} \qquad \text{Equation 4.24}$$

$$f(t) = C_5\, e^{\left(\frac{-C_6}{T}\right)} \qquad \text{Equation 4.25}$$

$$V(T) = C_1 \left[1 + C_{19} \left(\frac{4}{27} - \frac{J_3^2}{J_2^3}\right) + C_{20} \frac{J_3}{J_2^{\frac{3}{2}}}\right] e^{\left(\frac{-C_2}{T}\right)} \qquad \text{Equation 4.26}$$

where $J_2 = \frac{1}{2}\left(\underline{\sigma}' - \underline{\alpha}\right)^2$ and $J_3 = \frac{1}{3}\left(\underline{\sigma}' - \underline{\alpha}\right)^3$. The function $Y(T)$ is the rate-independent yield stress, $f(t)$ determines when the rate-dependence affects initial yielding, and $V(T)$ determines the magnitude of rate-dependence on yielding. The parameters C_1 through C_6 are material parameters determined experimentally. C_a and C_b are the material parameters that relate the stress state to the rate-independent yield stress, and C_{19} and C_{20} relate the stress state to the magnitude of rate-dependence on yielding.

The co-rotational rate of the kinematic hardening, $\overset{0}{\underline{\alpha}}$, and the material time derivative of isotropic hardening, $\dot{R}$, are cast in a hardening-recovery format that includes dynamic and static recovery as

$$\overset{0}{\underline{\alpha}} = \left\{h(T)\underline{D}^p - \left[\sqrt{\frac{2}{3}}\, r_d\,(T)\left\|\underline{D}^p\right\| + r_s\,(T)\right]\left\|\underline{\alpha}\right\|\underline{\alpha}\right\}\left(\frac{DCS_0}{DCS}\right)^z, \qquad \text{Equation 4.27}$$

$$\dot{R} = \left\{H(T)\underline{D}^p - \left[\sqrt{\frac{2}{3}}\, R_d\,(T)\left\|\underline{D}^p\right\| + R_s\,(T)\right]R^2\right\}\left(\frac{DCS_0}{DCS}\right)^z. \qquad \text{Equation 4.28}$$

DCS_0, DCS, and z are parameters to capture the microstructure effect of grain size as mentioned earlier. The scalar functions of temperature $r_d\,(T)$ and $R_d\,(T)$

describe dynamic recovery, $r_s\,(T)$ and $R_s\,(T)$ describe thermal (static) recovery, and $h(T)$ and $H(T)$ represent the anisotropic and isotropic modulus. These six functions account for deformation-induced anisotropy caused by texture and dislocation substructures. They are modified versions of those found in Miller et al. (1995) and Horstemeyer et al. (1995) where the hardening and recovery terms had a stress state dependence. For this thesis, the stress state dependence has been placed on the yield terms leaving the hardening and recovery terms as

$$r_d(T) = C_7 e^{\left(\frac{-C_8}{T}\right)}, \qquad \text{Equation 4.29}$$

$$R_d(T) = C_{13} e^{\left(\frac{-C_{14}}{T}\right)}, \qquad \text{Equation 4.30}$$

$$r_s(T) = C_{11}\, e^{\left(\frac{-C_{12}}{T}\right)}, \qquad \text{Equation 4.31}$$

$$R_s(T) = C_{17}\, e^{\left(\frac{-C_{18}}{T}\right)}, \qquad \text{Equation 4.32}$$

$$h(T) = C_9 - C_{10}T, \qquad \text{Equation 4.33}$$

$$H(T) = C_{15} - C_{16}T, \qquad \text{Equation 4.34}$$

C_7 through C_{12} are the material plasticity parameters related to the kinematic hardening and recovery terms, C_{13} through C_{18} are the material plasticity parameters related to the isotropic hardening and recovery terms.

Damage was introduced in the kinematics into Equation 4.7 as the void volume fraction. Different damage evolution rules can be included into the framework because of its inclusion as an ISV. Equations 4.18, 4.19, and 4.20 describe the void nucleation, growth, and coalescence, respectively. The models

come together within the plasticity framework where the time derivative of damage, $\dot{\Phi}$, is expressed as

$$\dot{\Phi} = \left(\dot{\Phi}_{particles} + \dot{\Phi}_{pores}\right)c + \left(\Phi_{particles} + \Phi_{pores}\right)\dot{c} \qquad \text{Equation 4.35}$$

where

$$\dot{\Phi}_{particles} = \dot{\eta}v + \eta\dot{v} \qquad \text{Equation 4.36}$$

$$\dot{\Phi}_{pores} = \left[\frac{1}{\left(1 - \Phi_{pores}\right)^{m}} - \left(1 - \Phi_{pores}\right)\right] \sinh\left[\frac{2\left(\frac{2\,V(T)}{Y(T)} - 1\right)}{\left(\frac{2\,V(T)}{Y(T)} + 1\right)} \frac{\sigma_H}{\sigma_{vm}}\right] \left\| \underline{D}^{p} \right\| \qquad \text{Equation 4.37}$$

$$\dot{c} = \left[Cd_1 + Cd_2(\dot{\eta}v + \eta\dot{v})\right] e^{(C_{ct}T)} \left(\frac{DCS_0}{DCS}\right)^{z} . \qquad \text{Equation 4.38}$$

The variables $\Phi_{particles}$ and Φ_{pores} represents void growth from particle debonding and fracture, and void growth from pores, respectively. Each has their corresponding time derivatives.

The integrated form of Equation 4.35 is used as the damage state. When damage, Φ, approaches unity, failure is assumed to have occurred. However, failure can be assumed to have occurred at a much smaller value. In practice, the total damage for failure should probably be less than 50% and is material dependent, but in most applications the damage will approach unity rapidly after it reaches a certain percentage.

The DMG model incorporates separate void nucleation, growth, and coalescence evolution equations in the BCJ ISV plasticity model to solve problems using the finite element method. The practicality of this model along with its physical basis makes it an ideal choice for use in the problems seen in this paper.

4.3.4 DMG Model Constants

The DMG material constants for AM60 magnesium alloy came from Horstemeyer et al. (2007) and the AM60 magnesium alloy simulations used the DMG model equations from that paper. The DMG constants for AZ31 magnesium alloy, A356 aluminum alloy, and 6061 aluminum alloy were obtained from experimental stress-strain data in tension and compression and strain rate-dependent nucleation data. However, coalescence data for those materials was not available, so they were estimated by matching the failure strain of the simulations incorporating predicted coalescence constants with the failure strain from experimental data. Coalescence data needs to be explicitly determined and implemented into these constants before they can be validated. The AZ31 magnesium alloy, A356 aluminum alloy, and 6061 aluminum alloy used the DMG model equations from this paper, and their DMG constants can be found in the appendix.

CHAPTER V

SHOCK WAVE AND STRUCTURAL RESPONSE MODELS

5.1 Introduction

The computational modeling used in this analysis consisted of two stages, shock wave modeling and structural response modeling. The shock wave modeling initially began as a planar wave with pressure values from the empirical data-based program ConWep. ConWep can model only planar waves, which did not accurately reflect the shock wave for our analysis, so the hydrocode CTH was used to capture the non-uniform loading. After the shock wave had been modeled, the pressure values were input into the finite element program Abaqus/Explicit. The modeling stages are discussed in detail in the following sections.

5.2 Shock Wave Model

Modeling of the shock wave was done in four phases with each successive phase adding a level of refinement not included in the previous phase. The first phase used a shock wave that was modeled as a planar wave using pressure values from ConWep. The next phase incorporated the computational hydrodynamic software CTH and modeled the shock wave as an open air blast using a reflection coefficient. The last two phases also used CTH to find the shock pressures but from flush buried explosives instead of explosives in open air. The pressure histories were initially multiplied by a reflection

coefficient to account for pressures when the shock wave encounters a structure. The last phase included a rigid body in the CTH simulations and the reflected pressures from the rigid body were calculated directly.

5.2.1 ConWep

ConWep is a program that calculates the blast effects, such as projectile penetration and shock wave strength, of conventional weapons using the equations and curves found in TM 5-855-1, *Design and Analysis of Hardened Structures to Conventional Weapons Effects* (1986). The equations found in TM 5-855-1 and used in ConWep were developed by Kingery and Bulmash (1984) to predict air blast parameters from spherical air bursts and hemispherical surface bursts. Those equations were found by curve-fitting high-order polynomial equations to experimental data from explosive test using charge weights from less than 1 kg to over 400,000 kg (Remennikov, 2003).

All air blast calculations used in ConWep are based on the explosive weight of TNT. For other types of explosives, ConWep uses the average of the equivalent weight factor for pressure and impulse to find an equivalent weight of TNT (see Chapter 2 for explanation). However, TNT was the only explosive used in these simulations. ConWep also uses an exponential decay function, Equation 5.1, for its pressure histories instead of the triangular histories proposed by TM 5-855-1 which should yield more accurate results because realistic pressure histories decay exponentially instead of linearly.

$$P(t) = P_{so}\left[1 - \frac{t - T_a}{T_0}\right] e^{\frac{-A(t - T_a)}{T_0}} \qquad \text{Equation 5.1}$$

where P(t) is the pressure at time t, P_{so} is the peak incident pressure, T_0 is the positive phase duration, A is the decay coefficient, and T_a is the arrival time. The negative phase of the pressure history is not taken into account which is a good approximation for close ranges. The same method is used for the reflected pressure histories. Additionally, the reflected pressures assume an infinitely large reflective surface which does not take into account clearing effects but yields conservative results (Hyde, 1988).

ConWep was used to make incident and reflected pressure histories for a hemispherical surface burst of a 3-kg TNT explosive at a distance of 46 cm as shown in Figure 5.1.

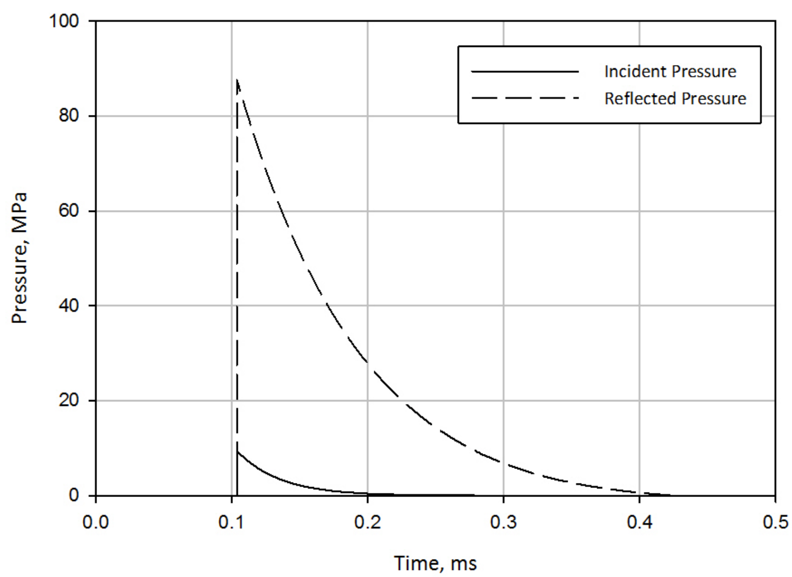

Figure 5.1 ConWep pressure history for 3 kg TNT explosive.

These pressures are applied to the structure as a planar wave, which is simply a uniform load varying with time. Because shock waves are not planar but non-uniformly distributed at the close ranges used, the hydrocode CTH was used to

capture the spatially- and temporally-varying pressure histories for the 3-kg TNT explosive.

5.2.2 CTH

To obtain pressure histories from shock waves with correct wave forms, explosive effects were investigated by numerical hydrocode simulations with the three-dimensional, Eulerian code CTH. CTH was designed to simulate shock wave propagation and material motion phenomena. Three series of CTH simulations were performed to determine the pressure histories representative of a mine blast: open air explosion multiplied by a reflection coefficient, flush buried explosion multiplied by a reflection coefficient, and flush buried explosion with directly measured reflected pressures.

Initial simulations modeled an explosion in air as seen in Figure 5.2. The incident pressure histories were determined for a 3 kg TNT explosive at a distance of 46 cm from the explosive. These pressure histories were then multiplied by a reflection coefficient to obtain the reflected pressures. As discussed in Chapter 2, these reflected pressures are the effective loading on a structure. However, an open air blast does not include pressures reflected off of the ground and underestimates the pressure histories. Therefore, the ground should be modeled in the simulations as well. The pressure history for the air blast is shown in Figure 5.3.

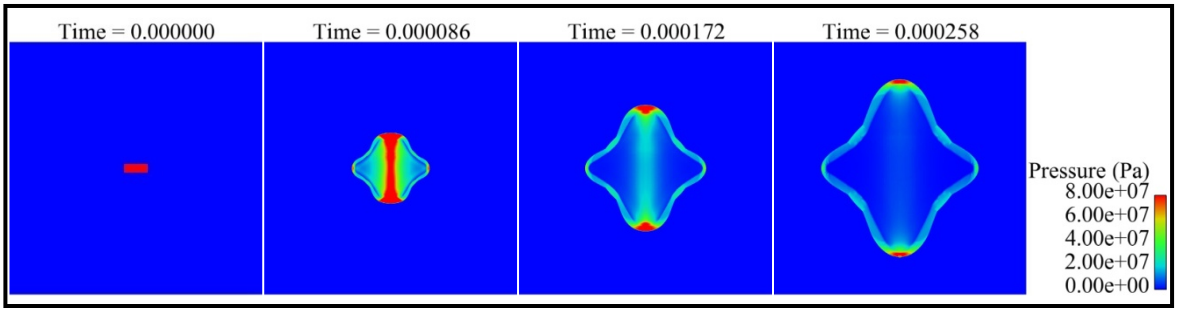

Figure 5.2 Explosion in air simulated with CTH.

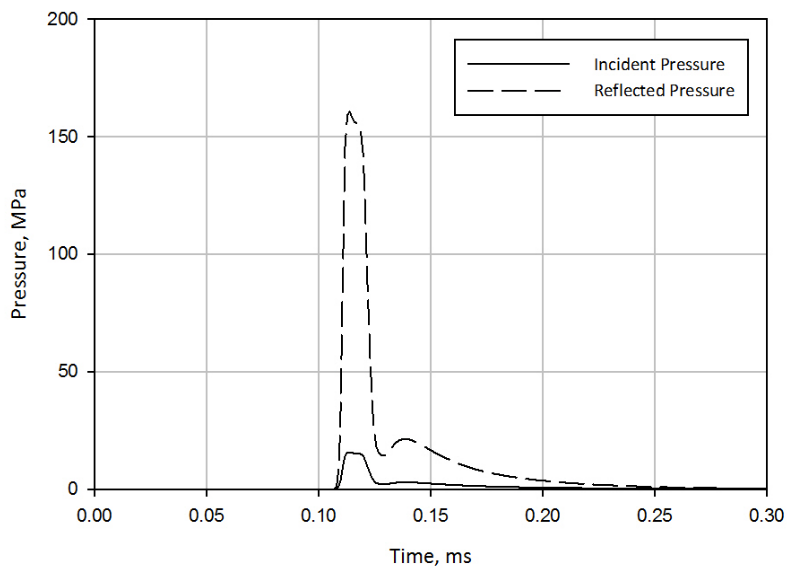

Figure 5.3 CTH pressure history for 3 kg TNT explosive in open air.

The next simulations found pressure histories from flush buried explosives which were also multiplied by reflection coefficients. Those simulations included the pressures reflected off of the ground, which make the results more realistic than the air blast simulations. However, the use of a reflection coefficient neglects relief effects when a shock wave encounters an obstacle and passes

around it. Figures 5.4 and 5.5 show the flush buried explosion and pressure history, respectively.

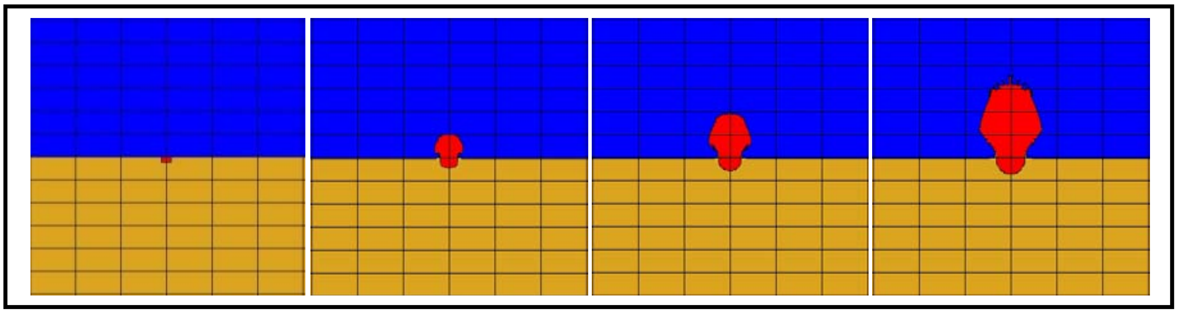

Figure 5.4 Flush buried explosive simulated with CTH.

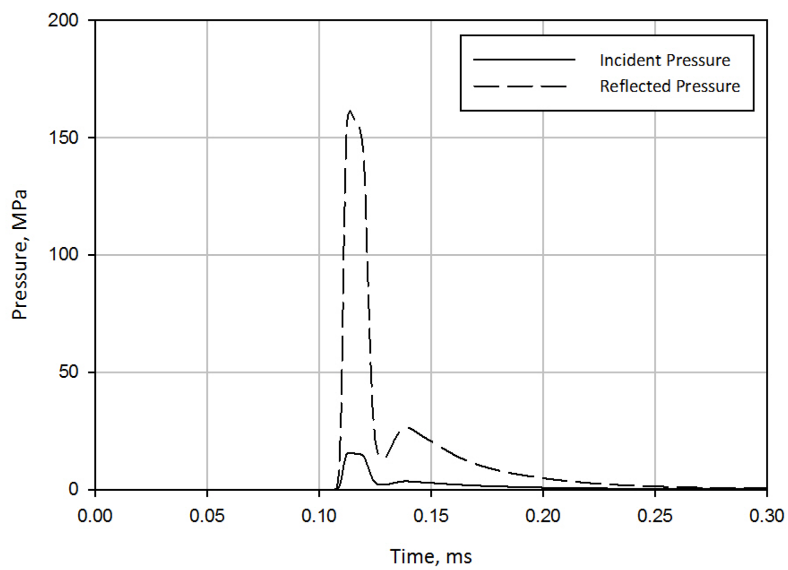

Figure 5.5 CTH pressure history for 3 kg TNT flush buried explosive.

The last and most comprehensive simulations used flush buried explosives with a rigid plate, and the actual reflected pressure histories were calculated. A more comprehensive soil model was also used in this simulation

as discussed in Chapter 4. This last simulation captured all of the major factors which would affect the magnitude of the shock waves. The pressure histories from this simulation are the ones used in the structural response model. Figures 5.6 and 5.7 show the flush buried explosion with rigid plate and pressure history, respectively.

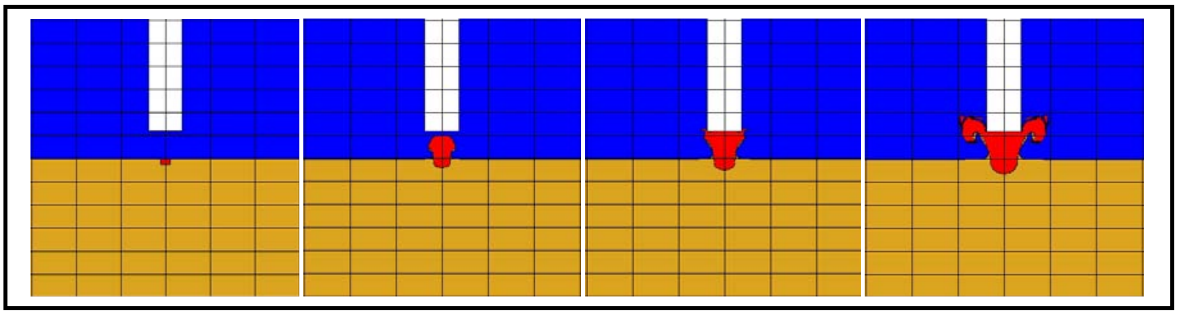

Figure 5.6 Flush buried explosive with rigid plate simulated with CTH.

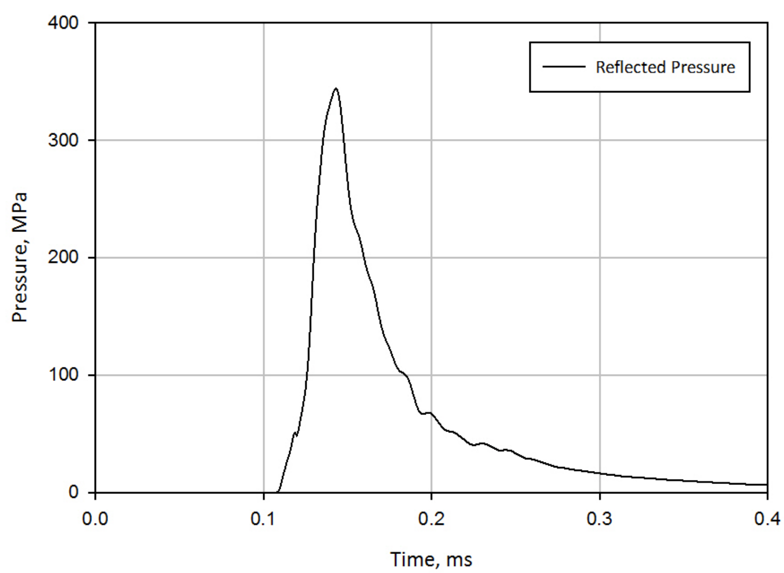

Figure 5.7 CTH reflected pressure history for 3 kg TNT flush buried explosive with rigid plate.

When comparing the four stages of shock wave modeling, the magnitudes of the peak pressures of the shock waves increased with each stage showing that the ConWep calculations were the least conservative for peak pressure predictions. However, ConWep over-predicted the impulse of the shock wave and was the least conservative in that regard. When comparing the CTH calculations, the first two simulations were nearly identical with the addition of the soil model providing only a small increase in the impulse, which was probably due to the soil used. As mentioned in Chapter 4, Kerley (2001) shows that dry sand is able to absorb more explosive energy than wet sand because the energy of dry sand is 50 - 100% greater than for wet sand at the same shock pressure. These initial calculations led up to the inclusion of wet sand that magnified the shock waves and a rigid plate that captured the reflected wave from the plate and relief effects of the shock wave going around the plate. The last CTH simulation yielded a peak pressure that more than doubled any of the previous simulations and had a greater impulse, also. This comparison is a good example of how each portion of the model, e.g., the type of soil, can have a large effect on the results.

The two-dimensional CTH simulations are all represented using a cylindrical coordinate system. This enables the simulations to be represented in half symmetry to save computational time. It also simplifies the pressure history data extraction. Tracers are placed along the rigid plate to capture the pressure history data at different points in space as shown in Figure 5.8.

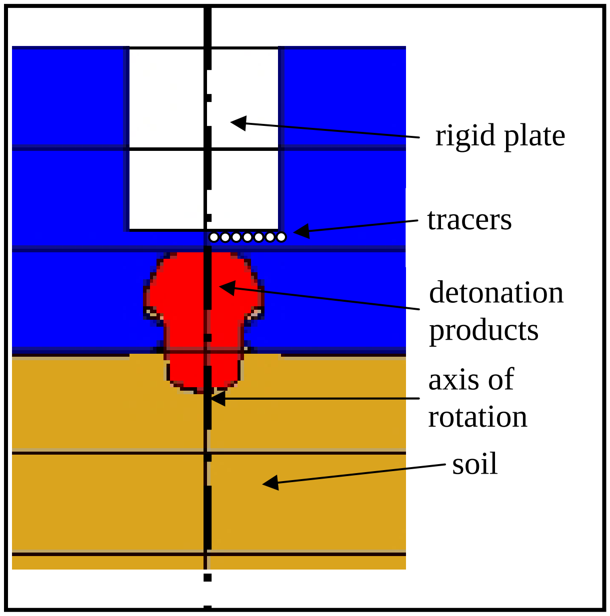

Figure 5.8 Location of tracers in CTH simulations.

Since the simulation is in the cylindrical coordinate system, the pressure histories are equivalent at equal distances from the axis of rotation. This will be useful when applying the pressure load to the plate in the structural response simulation as described in the next section.

After the initial structural response models were performed, it was observed that the thickest plates were not failing. So, the amount of TNT was scaled up until the desired failure was achieved. The amount of TNT used in the final models was 15 kg, and the pressure histories for the center of the 15 kg blast are shown in Figure 5.9. The peak pressures for the 15 kg explosive were almost three times as much as the pressures for the 3 kg explosives with comparable increases for the impulse as well.

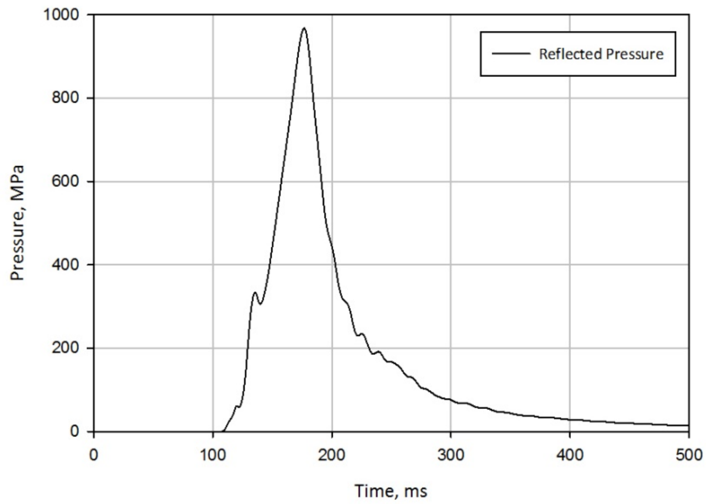

Figure 5.9 CTH reflected pressure history for 15 kg TNT flush buried explosive with rigid plate.

5.3 Structural Response Model

Abaqus/Explicit, a commercially available finite element analysis software package, was used to model the structural response of the plate during the shock wave impact. The structural response model was created to represent the response of a military Humvee floor pan to an explosive device blast. Pressure histories from CTH are applied to the plate using an Abaqus/Explicit VDLOAD subroutine.

The plate geometry used in the simulations was square with side lengths of 560 mm using 8-noded brick elements with reduced integration. These dimensions are approximately the dimensions of the floor pan used in the military Humvee. The Abaqus/Explicit input file can be found in the appendix for reference. The thickness of the plate is determined by the simulation corresponding to the DOE matrix as discussed in Chapter 3. The 10 mm and 30 mm plates contained approximately 90,000 elements, and the 50 mm and 75 mm

plates contained approximately 175,000 and 300,000 elements, respectively. Figure 5.10 shows the plate geometry for the 30 mm plate with the sacrificial element boundary conditions discussed next.

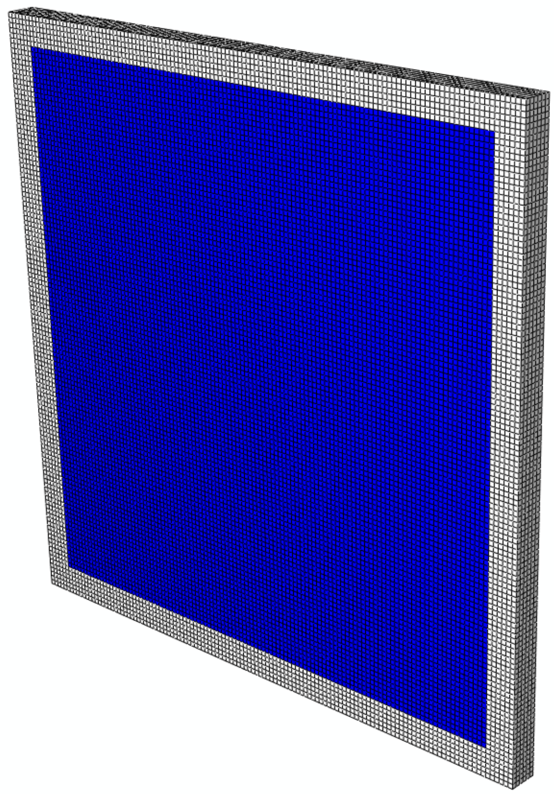

Figure 5.10 30mm plate geometry.

5.3.1 Boundary Conditions

Boundary conditions were found to have a significant effect on the predicted failure response of the model. Several different types of boundary conditions were developed and tested to reduce the likelihood that the boundary conditions would adversely affect the simulation results. Because clamped and pinned-like boundary conditions were of interest for the DOE analysis, suitable

boundary conditions were needed to capture the clamped and pinned-like behavior. Several types of boundary conditions were tested: constraining all of the degrees of freedom (DOF) or constraining the displacement DOF on one row of nodes along each side; constraining all of the DOF on all of the rows along each side; modeling each side as a rigid body with a reference node that governed the DOF; and modeling each side as a rigid body with a reference node and the addition of sacrificial elements.

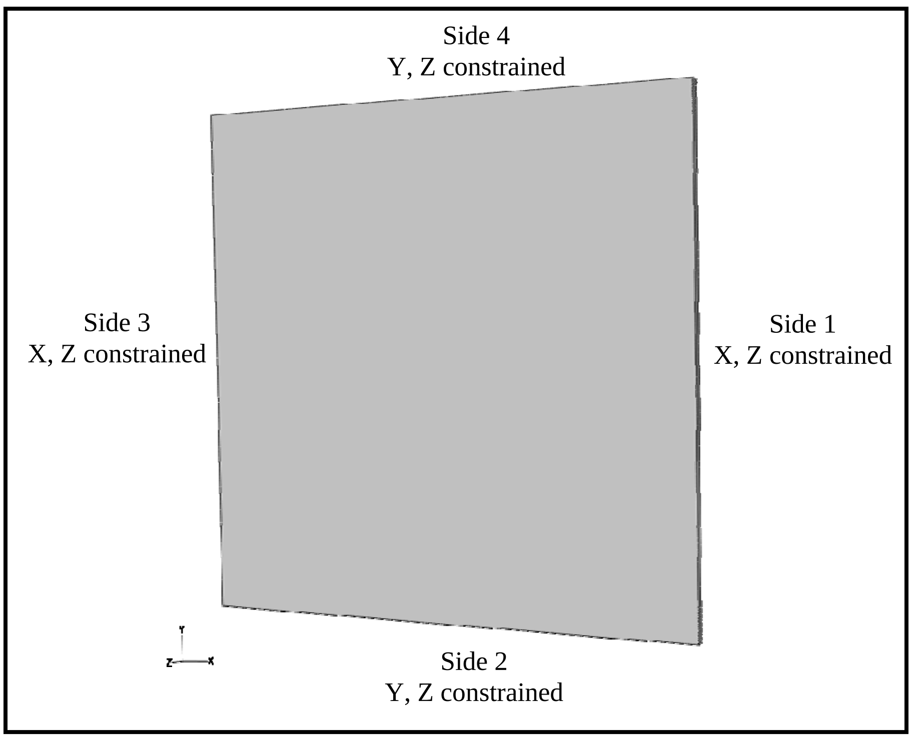

Figure 5.11 Simple pinned-like boundary conditions.

Modeling the simple pinned-like boundary condition proved difficult because these boundaries, where only one row of nodes was constrained on each side, yielded poor results. As shown in Figure 5.11, sides 1 and 3 were

constrained in the X and Z directions, and sides 2 and 4 were constrained in the Y and Z directions. None of the rotations were constrained. With this configuration, stress concentrations occurred around the constrained nodes and resulted in excessive hourglassing in the model as seen in Figure 5.12. Pinning all of the rows of nodes on the boundary leads to a clamped boundary condition where a pinned-like boundary condition is required.

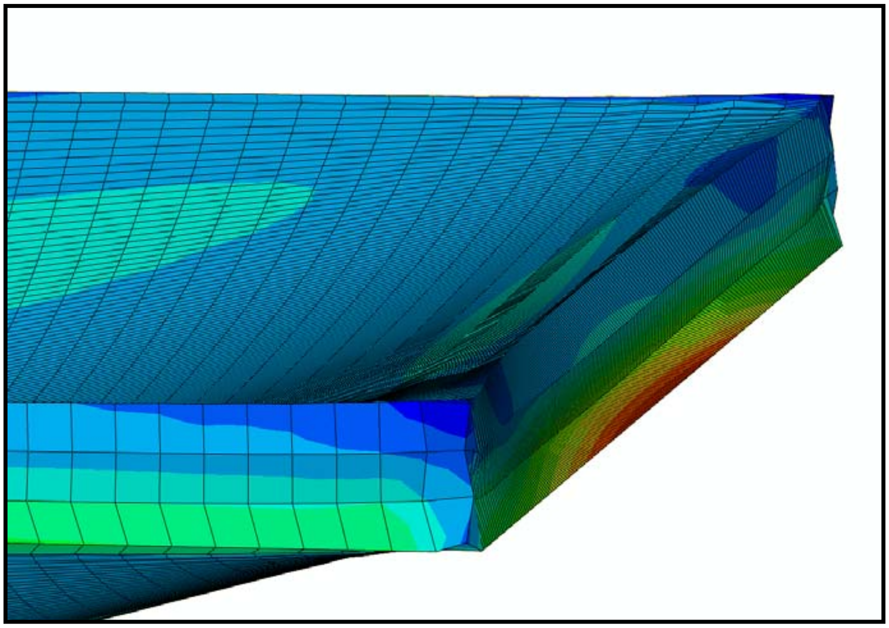

Figure 5.12 Problems with the simple pinned-like boundary condition.

To make a suitable pinned-like condition around the plate, every row of nodes on each side (except the nodes on each corner, which kept each side's boundary condition separated to allow rotation) was modeled as a rigid body governed by a reference node. This boundary setup allows the rotation of the rigid body, or plate edges, around the reference node, which is not attached to

the body. So, each side of the plate acts collectively as a rigid body with its rotation governed by the reference node. For the clamped boundary condition, the X, Y, and Z displacements and rotations at the reference node were constrained for each side face, which prevented the sides from translating and rotating. For the pinned-like boundary condition, the X, Y, and Z displacements were constrained for each side face, but the rotations were not. Figure 5.13 shows the location of the reference nodes for each side face of the plate.

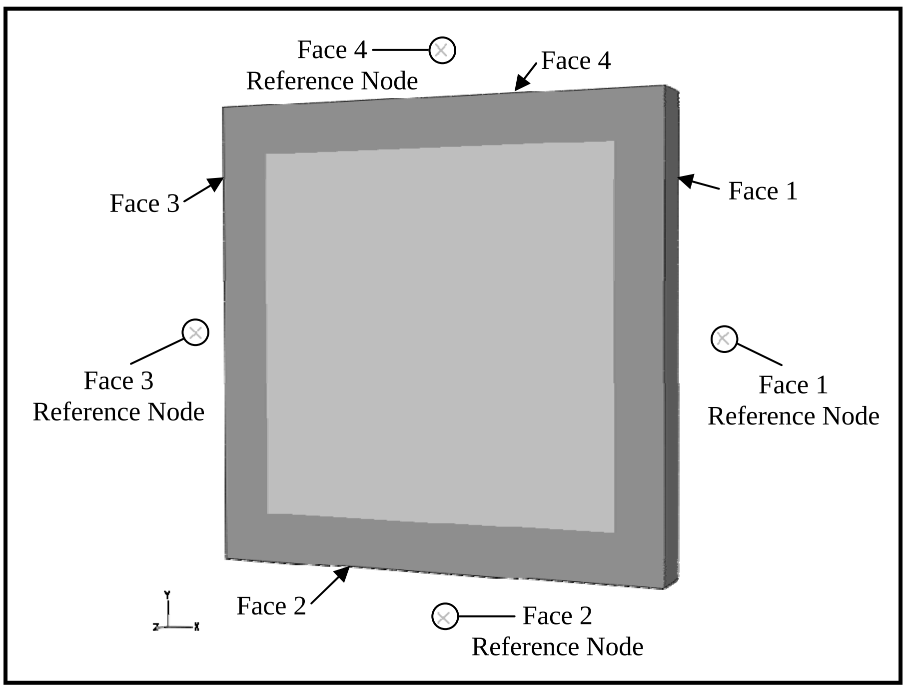

Figure 5.13 Boundary conditions using a reference node.

However, the rigid body boundaries crushed elements along the edges, which was not desirable, so the boundary conditions used in the final simulations contained sacrificial elements that were more compliant than the plate material and contained the boundary nodes. The width of the section of boundary

elements on each side was equal to the thickness of the plate. Identical to the last boundary conditions tested, a reference node was created on each that governed whether or not there could be rotation on that side. This type of boundary for the 75 mm plate can be seen in Figure 5.14.

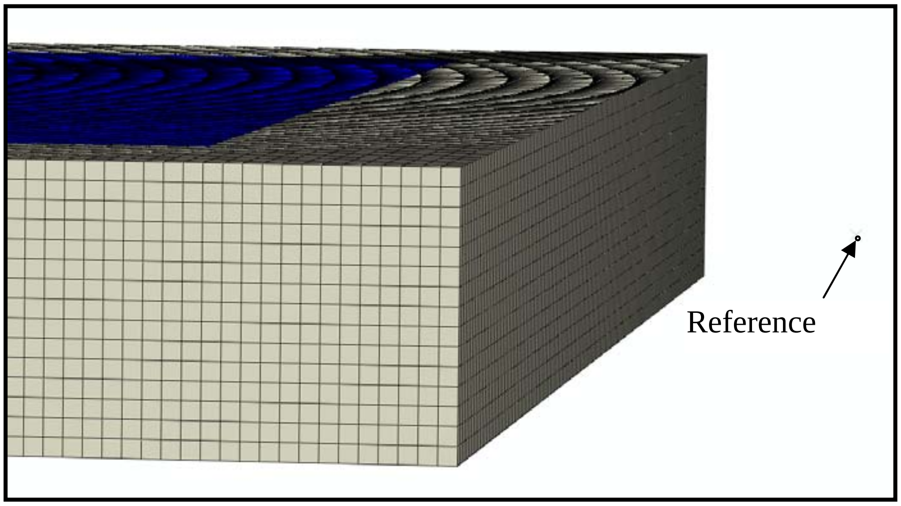

Figure 5.14 Boundary condition with sacrificial elements and reference node.

5.3.2 Abaqus VDLOAD Subroutine

To apply the time histories from the CTH simulations to the model, an Abaqus/Explicit user distributed load subroutine (VDLOAD) was written. The user load functionality allowed representation of the transient and spatial variation of the blast loads. The subroutine used the CTH tracer data and coordinates to determine the pressures to apply to each point. Linear interpolation was used to determine pressures in areas between tracer element data locations. The linear interpolation equation used in the VDLOAD is shown by

$$P(x) = P_0 + (x - x_0)\left(\frac{P_1 - P_0}{x_1 - x_0}\right) \qquad \text{Equation 5.2}$$

where P(x) is the pressure at x, and P_0, P_1, x_0, and x_1 are the pressures and positions of the data at the tracer locations as illustrated in Figure 5.15.

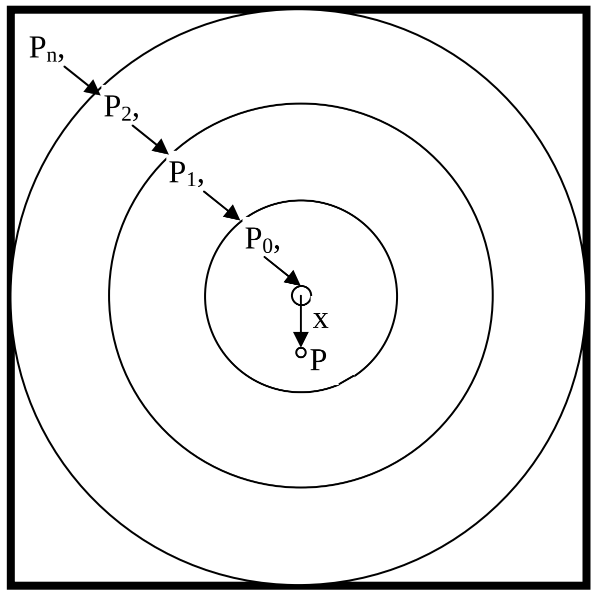

Figure 5.15 Illustration of linear interpolation used in Abaqus/Explicit VDLOAD.

The VDLOAD used a similar linear interpolation function to determine pressure values at times between the times pressures were recorded in CTH. The VDLOAD can be found in the appendix. Using the geometry and pressure loads determined here, the 16 DOE calculations were performed to determined the effects of the five parameters on energy absorption at first element failure. These results are discussed in the next chapter.

CHAPTER VI

INTERPRETATION OF RESULTS AND DISCUSSION

6.1 Introduction

The simulations were designed and carried out according to the Taguchi method described in Chapter 3. Energy absorption of the plate was the response investigated and was calculated as strain energy at first element failure. Sixteen simulations were performed and energy absorption was taken as the strain energy at first element failure. The effects of the parameters are discussed first followed by a discussion of the macroscale response of the plate. These energy absorption results can be seen in Table 6.1.

6.2 Discussion of Parameter Effects

The material response, energy absorbed at first element failure, for each level was analyzed using the Taguchi method. Table 6.1 shows the energy absorbed by first element failure for each of the 16 simulations. Those results were placed in the statistical software, Minitab 15 (2007), and analyzed to determine how the parameters affected the response. Table 6.2 shows the parameters according to their integer level numbers and Table 6.3 shows the energy absorption means for each level. The level numbers correspond to a particular object in a parameter in order to aid in graphical representations, e.g., AM60 is the level 1 object for the parameter material, 30mm is the level 2 object

for the parameter thickness, etc. Figure 6.1 shows the energy absorption means in graphical form.

Table 6.1 L_{16} orthogonal array showing the parameters.

Calculation	Material	Thickness	Temperature	Damage Distribution	Boundary Conditions	Strain Energy at 1st Element Failure (J)
1	AM60 Mg	10 mm	323 K	100% St. Dev.	Clamped	35077
2	AM60 Mg	30 mm	303 K	60% St. Dev.	Clamped	125072
3	AM60 Mg	50 mm	293 K	30% St. Dev.	Pinned-like	163126
4	AM60 Mg	75 mm	273 K	0% St. Dev.	Pinned-like	141904
5	AZ31 Mg	10 mm	303 K	30% St. Dev.	Pinned-like	24151
6	AZ31 Mg	30 mm	323 K	0% St. Dev.	Pinned-like	66684
7	AZ31 Mg	50 mm	273 K	100% St. Dev.	Clamped	48616
8	AZ31 Mg	75 mm	293 K	60% St. Dev.	Clamped	148882
9	A356 Al	10 mm	293 K	0% St. Dev.	Clamped	53211
10	A356 Al	30 mm	273 K	30% St. Dev.	Clamped	29625
11	A356 Al	50 mm	323 K	60% St. Dev.	Pinned-like	35444
12	A356 Al	75 mm	303 K	100% St. Dev.	Pinned-like	50300
13	6061 Al	10 mm	273 K	60% St. Dev.	Pinned-like	102328
14	6061 Al	30 mm	293 K	100% St. Dev.	Pinned-like	168227
15	6061 Al	50 mm	303 K	0% St. Dev.	Clamped	298742
16	6061 Al	75 mm	323 K	30% St. Dev.	Clamped	308755

Table 6.2   Integer representation of parameters as levels.

Level	Material	Thickness	Temperature	Damage Distribution	Boundary Conditions
1	AM60 Mg	10 mm	323 K	100 percent St. Dev.	Clamped
2	AZ31 Mg	30 mm	303 K	60 percent St. Dev.	Pinned-like
3	A356 Al	50 mm	293 K	30 percent St. Dev.	
4	6061 Al	75 mm	273 K	0 percent St. Dev.	

Table 6.3 Response table for means in joules.

Level	Material	Thickness	Temperature	Damage Distribution	Boundary Conditions
1	116295	53692	111490	75555	130998
2	72083	97402	124566	102932	94021
3	42145	136482	133362	131414	
4	219513	162460	80618	140135	
Delta	177368	108769	52743	64580	36977
Rank	1	2	4	3	5

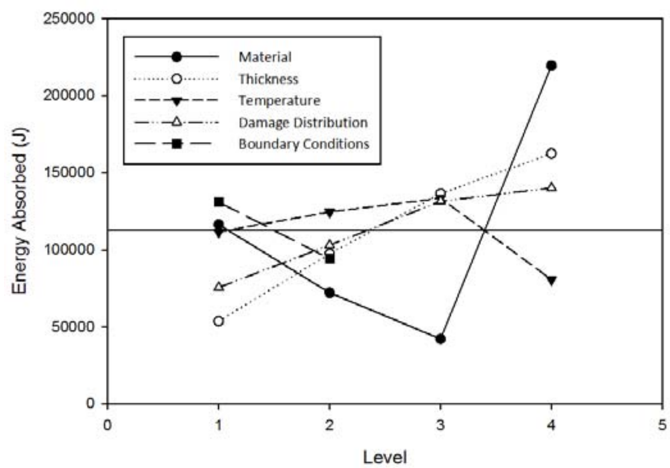

Figure 6.1 The effects of the parameters on energy absorbed by the plate.

In Table 6.3, the energy absorption means are shown along with the differences between the highest values to the lowest values called delta. The rankings for the influences of the parameters on the energy absorption were based on the deltas and are also shown. Material and thickness were primary influences followed by damage distribution, temperature, and boundary conditions. To normalize the influences of the parameters on the energy absorption, each delta was divided by the largest delta (from the parameter

material), which gives relative influences of each parameter as shown in Figure 6.2.

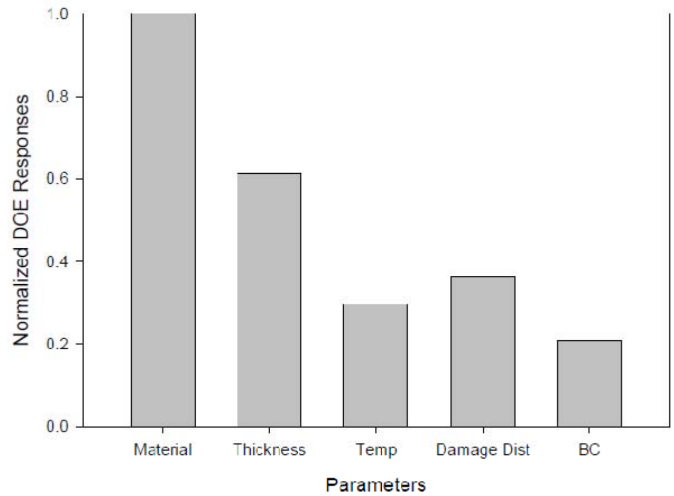

Figure 6.2 Normalized parameter influences on energy absorption.

6.2.1 Effects of Material

The material used for the plate was the largest influence on the energy absorption of the plate. The A356 aluminum absorbed the least amount of energy followed by the AZ31 magnesium, AM60 magnesium, and 6061 aluminum. The 6061 aluminum alloy had a significantly higher von Mises stress at failure than the other materials and one of the highest failure strains, which may be the reason that material absorbed the most energy. If other materials had been compared that had similar strengths and failure strains, the thickness should have a dominant influence. Table 6.4 shows the average von Mises stress for the first failed element of the four simulations for each material. The approximate failure strain in tension for each material is also included. From Table 6.4, it can be seen that the A356 aluminum alloy had the second highest

average von Mises stress at failure, but it was the worst performing material out of the four materials. This is likely due to the fact that the A356 had a significantly lower failure strain than the other materials. So, the best combination of high stress and strain to failure should lead to better performance of a material, which was the case for the 6061 aluminum alloy.

Table 6.4 Average Mises stress at failure and failure strains.

Material	Avg. Mises Stress at Failure (MPa)	Approximate Failure Strain in Tension
AM60 Magnesium	332	0.10
AZ31 Magnesium	286	0.14
A356 Aluminum	382	0.07
6061 Aluminum	443	0.13

6.2.2 Effects of Thickness

The plate thickness was another primary influence on the energy absorption. Though this was expected to be the main influence, the energy absorption due to the thickness was significantly less than that due to the material. As noted before, this may not be the case if different materials with more similar yield strengths and failure strains are used. Nevertheless, the response followed the expected trend as energy absorption increased with increased thickness.

6.2.3 Effects of Temperature

Temperature was a tertiary influence on the plate response. However, only the AM60 had temperature dependant stress-strain characterization. Better

characterization of the other materials for temperature dependence should increase this influence on the response. Therefore, the AM60 alone probably determined the effects of the temperature. The energy absorption gradually increases as temperature decreases as expected, except for 273 K where the mean energy absorbed is lower than all of the other temperatures. The energy absorption was expected to increase as the temperature increased because metals are weaker at higher temperatures. Characterization of the AZ31, A356, and 6061 alloys for temperature-dependence should make the energy absorption follow the expected trend because the effects of the temperature would be averaged over four materials instead of the AM60 only. The mean energy absorbed at first element failure for each temperature can be seen in Table 6.5.

Table 6.5 Response table for temperature means.

Temperature	Energy at First Element Failure
323 K	111490
303 K	124566
293 K	133362
273 K	80618
Difference	52743

6.2.4 Effects of Damage Distribution

The damage distribution in the plate was a secondary influence on the energy absorbed. It also behaved as expected since the ability to absorb energy decreased as the standard deviation of the damage distribution increased. As discussed in Chapter 3, the damage distribution with the highest standard deviation should absorb less energy because there are areas of concentrated

damage due to the non-uniformity of the distribution. This hypothesis was proven correct, as shown in Table 6.6.

Table 6.6 Response table for damage distribution means.

Damage Distribution	Energy at First Element Failure
100 percent St. Dev.	75555
60 percent St. Dev.	102932
30 percent St. Dev.	131414
0 percent St. Dev.	140135
Difference	64580

6.2.5 Effects of Boundary Conditions

The pinned-like boundary condition was expected to allow the absorption of more energy than the clamped boundary condition because it allows more plate movement. However, the boundary conditions influenced the response opposite of what was expected, but its influence was minor compared to the other parameters. Due to the high-rate nature of the shock wave impact, the effects of the boundary conditions are lessened from the lack of time allowed for the stress wave to travel to the boundaries. The mean energy at first element failure for all of the simulations for the pinned-like and clamped is 130,998 J and 94,021 J, respectively, which is the smallest difference for all of the parameters. If more simulations are to be run, this parameter could probably be left out and another parameter studied.

6.3 Discussion of Macroscale Effects

As discussed previously, the material was the primary influence on the energy absorbed in the plate. From observation of the simulations, it also

determined the type of failure that occurred after the shock wave impacted the plate. Each of the magnesium plates failed near the boundaries from either tearing away from the boundaries or from a compressive failure at the boundary in the case of some of the bigger plates. In all of the aluminum plates, tensile failure occurred in center of the plates resembling spalling. Some of these failures are shown in the following figures. All of the simulation figures can be found in the appendix for reference.

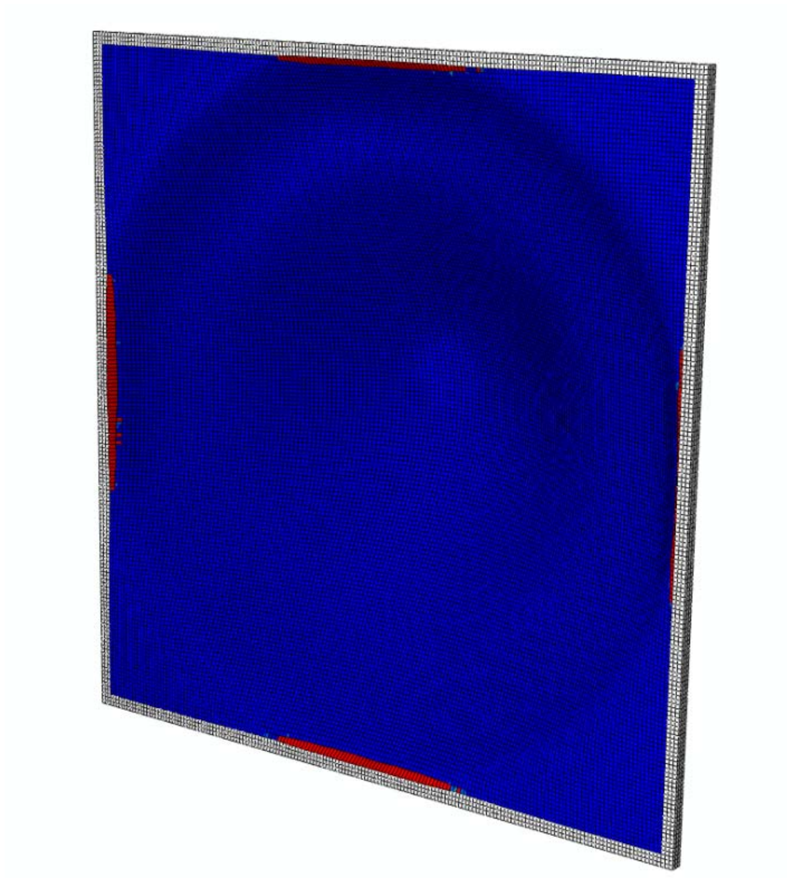

Figure 6.3 Tearing at the boundaries in magnesium plate.

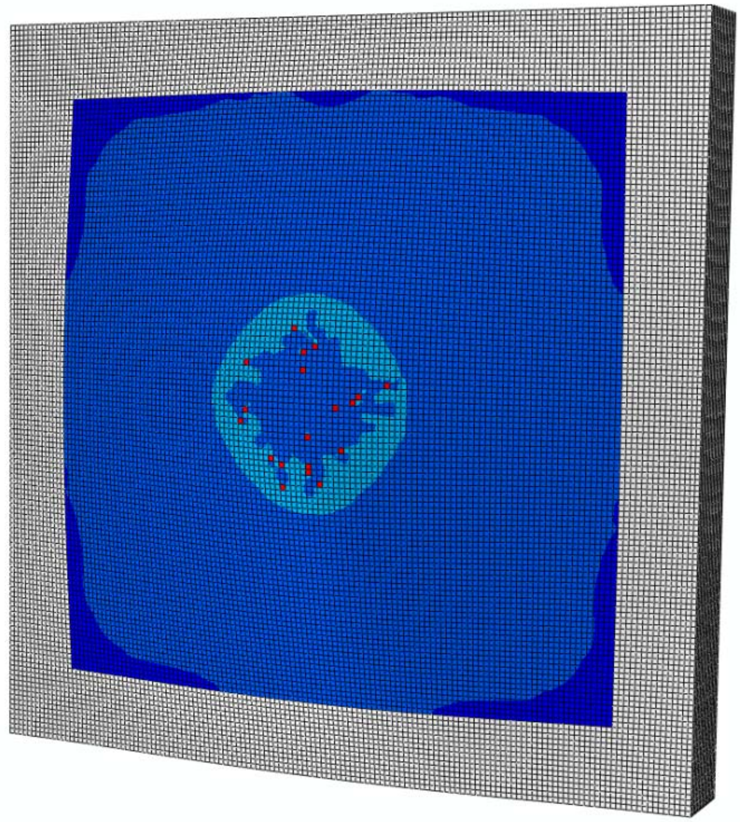

Figure 6.4   Spalling in aluminum plate.

The other primary influence, thickness, was the determining factor in the deflection of the plate. If deflection was the response being investigated, thickness would have likely been the dominating influence for that response, as shown by Table 6.7. The other factors did not seem to have an affect on the type of failure or any other responses that could be observed.

Table 6.7 Mean deflection at first element failure.

Thickness	Mean Deflection at 1st Element Failure (mm)
10 mm	102
30 mm	76
50 mm	56
75 mm	48

6.4 Conclusions and Recommendations

A design of experiments approach by Taguchi was used to evaluate the effects of five parameters on the energy absorption of a plate during a shock wave impact. The five parameters were material, thickness, temperature, damage distribution, and boundary conditions. The simulations were performed using finite element software implementing the DMG model.

The analysis indicated that the material and thickness were primary influences followed by damage distribution, temperature, and boundary conditions. The simulations showed that the 6061 aluminum should absorb the most energy followed by AM60 magnesium, AZ31 magnesium, and A356 aluminum. Energy absorption increased as thickness increased as expected. The damage distributions also followed the expected trend as energy absorption decreased as the standard deviation of the distribution increased. However, temperature did not behave as expected, which is probably due to incomplete characterization of the materials for the DMG model. Boundary conditions did not have a significant effect on the response.

Upon completion of this analysis, the main recommendation for improvement of this analysis and for other simulations using the DMG model is to

accurately and completely characterize the DMG material parameters that are being used in the model. The DMG material parameters used in this analysis were only fit for tension and compression stress states and the AZ31, A356, and 6061 have no temperature dependence. It would also be helpful to have experimental results to compare against the deflection data for verification.

In summary, the phenomenological aspects of the DMG model and the use of the DOE method combine to form a powerful tool to optimize components based on the physical aspects of a material. From this analysis, it can be seen that the material and thickness play major roles in the energy absorption of a plate. Though that was obvious before the analysis, the use of a phenomenological model has shown the failure modes for this type of blast loading for aluminum and magnesium, tearing and spalling, respectively. Furthermore, this framework would be even more useful for very specific problems, e.g., minimizing weight of a floor pan armor plate to eliminate spalling from a 5 kg TNT blast. For those specific problems, this type of analysis would be ideal to determine the material, its thickness, and the manufacturing processes to reduce initial damage while lowering the weight of component. Therefore, the DOE technique and DMG damage model are recommended for use in blast mitigation component analyses.

REFERENCES

Abaqus/Explicit. Vers. 6.9. Providence, RI: Simulia, 2009. Computer software.

Baker, W. E. *Explosions in Air,*. Austin: University of Texas, 1973.

Bammann, D.J., and E.C. Aifantis. "A Damage Model for Ductile Metals." *Nuclear Engineering and Design* 116 (1989): 355-62.

Bammann, D.J., M.L. Chiesa, M.F. Horstemeyer, and L.I. Weingarten. "Failure in Ductile Materials Using Finite Element Methods." *Structural Crashworthiness and Failure*. Ed. Norman Jones and Tomasz Wierzbicki. London: Elsevier Science, 1993.

Bellamy, Ronald F., Russ Zajtchuk, and Teresa M. Buescher. *Conventional Warfare: Ballistic, Blast, and Burn Injuries*. Washington, D.C.: Walter Reed Army Institute of Research, Walter Reed Army Medical Center, 1991.

Box, G.E., W.G. Hunter, and J.S. Hunter. *Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building*. New York: Wiley, 1978.

Braid, M. *Experimental Investigation and Analysis of the Effects of Anti-Personnel Landmine Blasts*. Thesis. Royal Military College of Canada, 2001. Defence R&D Canada, 2001.

Brode, H.L. *Numerical Solutions of Spherical Blast Waves*. RM-1363-AEC. Santa Monica, CA: Rand Corporation, 1954.

Camacho, G.T. and Ortiz, M. “Computational modeling of impact damage in brittle materials.” *International Journal of Solids and Structures* 33 (1996): 2899–2938.

Cirak, F., Ortiz, M. and Pandolfi, A. “A cohesive approach to thin-shell fracture and fragmentation.” *Computer Methods in Applied Mechanics and Engineering* 194 (2005)**:** 2604–18.

ConWep. Vers. 2.1.0.8. Vicksburg, MS: USAE Engineer Research and Development Center, 2005. Computer software.

Cooper, Paul W. *Explosives Engineering*. New York, N.Y.: VCH, 1996.

Cox, T.B., and J.R. Low Jr. "An Investigation of the Plastic Fracture of AISI 4340 and 18 Nickel-200 Grade Maraging Steels." *Metallurgical and Materials Transactions B* 5 (1974): 1457-470.

CTH. Vers. 8.1. Albuquerque, NM: Sandia National Laboratories, 2007. Computer software.

Davison, L., A.L. Stevens, and M.E. Kipp. "Theory of Spall Damage Accumulation in Ductile Metals." *Journal of the Mechanics and Physics of Solids* 25 (1977): 11-28.

Detail Specification Armor Plate, Magnesium, AZ31B, Applique. Tech. no. MIL-DTL-32333. U.S. Army, 2009. New Magnesium Armor Plate MIL Spec. Magnesium Elektron. Web. 20 Jan. 2010.

Dharmasena, K.P., H.N.G. Wadley, Z. Xue, and J. Hutchinson. "Mechanical Response of Metallic Honeycomb Sandwich Panel Structures to High-intensity Dynamic Loading." *International Journal of Impact Engineering* 35 (2008): 1063-074.

Dobratz, B.M., and P.C. Crawford. *LLNL Explosives Handbook*. UCRL-52997. Lawrence Livermore National Laboratory Report, 1985.

Drugan, W.J. “Dynamic fragmentation of brittle materials: analytical mechanics-based models.” *Journal of the Mechanics and Physics of Solids* 49 (2001): 1181–1208.

Espinosa, H.D., Zavattieri, P.D. and Dwivedi, S.K. “A finite deformation continuum/discrete model for the description of fragmentation and damage in brittle materials.” *Journal of the Mechanics and Physics of Solids* 46 (1998): 1909–1942.

Garber, R., I.M. Berstein, and A.W. Thompson. "Effect of Hydrogen on Ductile Fracture of Spheroidized Steel." *Scripta Metallurgica* 10 (1976): 341.

Glenn, L.A. and Chudnovsky, A. “Strain-energy effects on dynamic fragmentation.” *Journal of Applied Physics* 59 (1986): 1379-1380.

Graboske, H.C. *A New EOS for Air*. UCID-16901. Lawrence Livermore National Laboratory Report, 1975.

Grady, D.E. “Local inertial effects in dynamic fragmentation.” *Journal of Applied Physics* 53 (1982): 322-325.

Grady, D.E. “Particle size statistics in dynamic fragmentation.” *Journal of Applied Physics* 68 (1990): 6099-6105.

Grady, D.E. and Kipp, M.E. “Geometric statistics and dynamic fragmentation.” *Journal of Applied Physics* 58 (1985): 1210-1222.

Henrych, J. *The Dynamics of Explosion and Its Use*. Amsterdam: Elsevier Scientific Pub., 1979.

Hertel Jr., E.S., and G.I. Kerley. *CTH EOS Package: Introductory Tutorial*. SAND98-0945. Sandia National Laboratories Report, 1998.

Horstemeyer, M.F. "Structural Analysis of a Submarine Using Statistical Design of Experiments." *Advances in Numerical Simulation Techniques for Penetration and Perforation of Solids: Presented at the 1993 ASME Winter Annual Meeting New Orleans, Louisiana November 28-December 3, 1993*. Ed. E. P. Chen and V. K. Luk. New York, N.Y.: American Society of Mechanical Engineers, 1993. 189-200.

Horstemeyer, M.F., and A.M. Gokhale. "A Void-crack Nucleation Model for Ductile Metals." *International Journal of Solids and Structures* 36 (1999): 31-47.

Horstemeyer, M.F., and D.L. McDowell. *Using Statistical Design of Experiments for Parameter Study of Crystal Plasticity Modeling Features under Different Strain Paths*. Rep. no. SAND96-8683. Sandia National Laboratory Report, 1997.

Horstemeyer, M.F., D. Oglesby, J. Fan, P.M. Gullett, H. El Kadiri, Y. Xue, C. Burton, K. Gall, B. Jelinek, M.K. Jones, S.G. Kim, E.B. Marin, D.L. McDowell, A. Oppedal, and N. Yang. *From Atoms to Autos: Designing a Mg Alloy Corvette Cradle by Employing Hierarchical Multiscale Microstructure-Property Models for Monotonic and Cyclic Loads*. MSU.CAVS.CMD.2007-R0001. 2007.

Horstemeyer, M.F., D.L. McDowell, and D.J. Bammann. "Torsional Softening and the Forming Limit Diagram." Ed. K. Chen. *Analysis of Autobody Stamping Technology.* Warrendale, Pa.: Society of Automotive Engineers, 1994.

Horstemeyer, M.F., D.L. McDowell, and R.D. McGinty. "Design of Experiments for Constitutive Model Selection: Application to Polycrystal Elastoviscoplasticity." *Modeling and Simulation in Materials Science and Engineering* 7 (1999): 253-73.

Horstemeyer, M.F., J. Lathrop, A.M. Gokhale, and M. Dighe. "Modeling Stress State Dependent Damage Evolution in a Cast Al-Si-Mg Aluminum Alloy." *Theoretical and Applied Fracture Mechanics* 10 (2000): 31-47.

Horstemeyer, M.F., S. Ramaswamy, and M. Negrete. "Using a Micromechanical Finite Element Parametric Study to Motivate a Phenomenological Macroscale Model for Void/crack Nucleation in Aluminum with a Hard Second Phase." *Mechanics of Materials* 35 (2003): 675-87.

Hyde, D. *User's Guide for Microcomputer Programs ConWep and FUNPRO – Applications of TM 5-855-1*. Vicksburg, MS: U.S. Army Engineer Waterways Experimental Station, 1988.

Kerley, G. I. *Numerical Modeling of Buried Mine Explosions*. ARL-CR-461. Aberdeen Proving Ground, MD: U.S. Army Research Laboratory, 2001.

Kerley, G.I. *On the Numerical Simulation of Buried Mine Explosions: Choosing Constitutive Models*. KTS05-3. Aberdeen Proving Ground, MD: U.S. Army Research Laboratory, 2005.

Kingery, C.N., and G. Bulmash. *Airblast Parameters from TNT Spherical Air Burst and Hemispherical Surface Burst*. ARBL-TR-02555. Aberdeen Proving Ground, MD: U.S. Army, 1984.

Lee, E.L., H.C. Hornig, and J.W. Kury. *Adiabatic Expansion of High Explosive Detonation Products*. UCRL-50422. Livermore, CA: Lawrence Radiation Laboratory, 1968.

Major, J.F., A. Makinde, P.D. Lee, B. Chamberlain, T. Scappaticci, and D. Richman. "The Lincoln Mark VIII Cast Aluminum Suspension Control Arm (Parallel Development)," Journal of Materials and Manufacturing 103 (1994): 635- 646.

McClintock, F.A. "A Criterion for Ductile Fracture by the Growth of Holes." *ASME Journal of Applied Mechanics* 35 (1968): 363-71.

McGlaun, J.M., F.J. Ziegler, S.L. Thompson, L.N. Kmetyk, and M.G. Elrick. *CTH - User's Manual and Input Instructions*. SAND88-0523. Sandia National Laboratories, 1988.

Miller, M.P., P.R. Dawson, and D.J. Bammann. "Reflecting Microstructural Evolution in Hardening Models for Polycrystalline Metals." *Simulation of Materials Processing: Theory, Methods and Applications : Proceedings of the Fifth International Conference on Numerical Methods in Industrial Forming Processes--NUMIFORM '95, Ithaca/New York/USA/18-21 June, 1995*. Ed. Shan-Fu Shen and Paul R. Dawson. Rotterdam: A.A. Balkema, 1995.

Miller, O., Freund, L.B. and Needleman, A. "Modeling and simulation of dynamic fragmentation in brittle materials." *International Journal of Fracture* 96 (1999): 101-125.

Minitab 15 Statistical Software. State College, PA: Minitab, 2007. Computer software.

Nedler, J.A., and Y. Lee. "Generalized Linear Models for the Analysis of Taguchi-type Experiments." Ed. Carlo Lauro. *Applied Stochastic Models and Data Analysis*. Chichester: Wiley-Interscience, 1991. 107-20.

Peace, Glen Stuart. *Taguchi Methods: A Hands-On Approach to Quality Engineering*. Reading, Mass. ; Don Mills, Ont.: Addison-Wesley, 1993.

Rankine, W.J.H. *Philosophical Transactions of the Royal Society* 160.277 (1870).

Remennikov, A.M. "A Review of Methods for Predicting Bomb Blast Effects on Buildings." *Journal of Battlefield Technology* 6: 5-10.

Rickman, D., and D. Murrell. "Development of an Improved Methodology for Predicting Airblast Pressure Relief on a Directly Loaded Wall." *Journal of Pressure Vessel Technology* 129 (2007): 195-204.

Rogers, H.C. "The Tensile Fracture of Ductile Metals." *Transactions of the Metallurgical Society of AIME* 218 (1960): 498-506.

Roy, R. K. *Design of Experiments Using the Taguchi Approach: 16 Steps to Product and Process Improvement.* New York, N.Y.: Wiley, 2001.

Sachs, R.G. *The Dependence of Blast on Ambient Pressure and Temperature*. BRL Report 466. Aberdeen Proving Ground, MD, 1944.

Smith, P. D., and J. G. Hetherington. *Blast and Ballistic Loading of Structures*. Oxford: Butterworth-Heinemann, 1994.

Structures to Resist the Effects of Accidental Explosions. UFC 3-340-02. Department of Defense, 1990.

Stutsman, R., B. Chamberlain, and M. Evans. "The Design of an Experiment to Choose an Aluminum Die Casting Alloy for Energy Absorbing Automotive Components." *SAE-SP-1164 No. 960159* (1996): 1-7.

The Design and Analysis of Hardened Structures to Conventional Weapons Effects Manual. TM 5-855-1. U.S. Army, 1986.

Trinh, K.T., and J. Gruda. *Failure Resistance of Thin Shells against Projectile Penetration*. Rep. no. SAND91-8486. Livermore, California: Sandia National Laboratories, 1991.

Tucker, M.T., M.F. Horstemeyer, W.R. Whittington, K.N. Solanki, and P.M. Gullett. "The Effect of Varying Strain Rates and Stress States on the Plasticity, Damage, and Fracture of Aluminum Alloys." Submitted for publication.

Wilkinson, C.R., and J.G. Anderson. *An Introduction to Detonation and Blast for the Non-Specialist*. DSTO-TN-0526. Edinburgh, South Australia: DSTO Systems Sciences Laboratory, 2003.

Yatom, H. and Ruppin, R. “Dynamic fragmentation model with internal damage.” *Journal of Applied Physics* 65 (1989): 112-116.

Yew, C.H. and Taylor, P.A. “A thermodynamic theory of dynamic fragmentation.” *International Journal of Impact Engineering* 15 (1994): 385-394.

Young, S.H. *An Optimization Technique Using the Finite Element Method and Orthogonal Arrays*. Thesis. Naval Postgraduate School, 1996. Monterey, CA.

Zhou, F., Molinari, J.F. and Ramesh, K.T. “Characteristic fragment size distributions in dynamic fragmentation.” *Applied Physics Letters* 88 (2006).

APPENDIX A

ABAQUS SIMULATION RESULTS

Please note that most of these figures are well after first element failure.

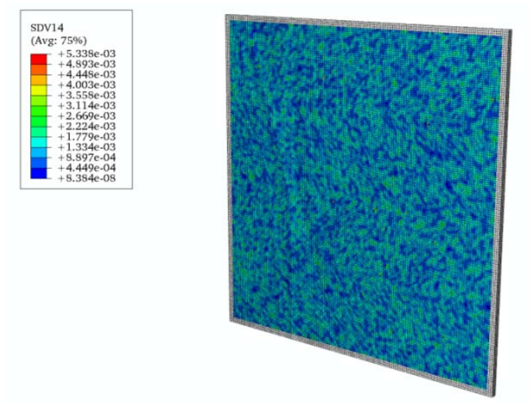

Figure A.1 Calculation 1 initial damage.

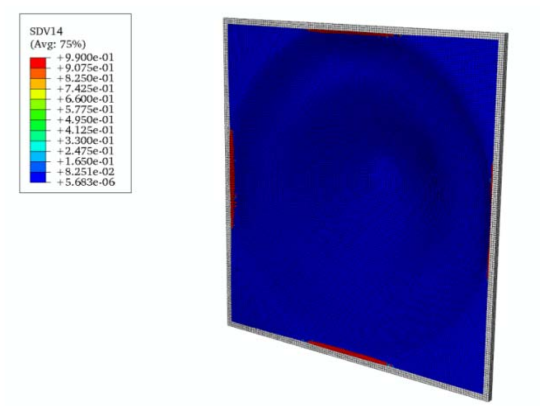

Figure A.2 Calculation 1 final damage front.

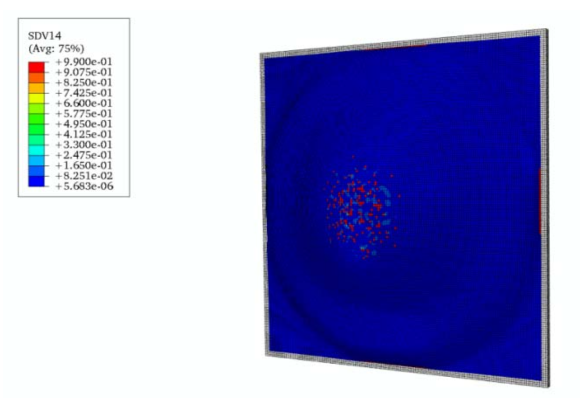

Figure A.3 Calculation 1 final damage back.

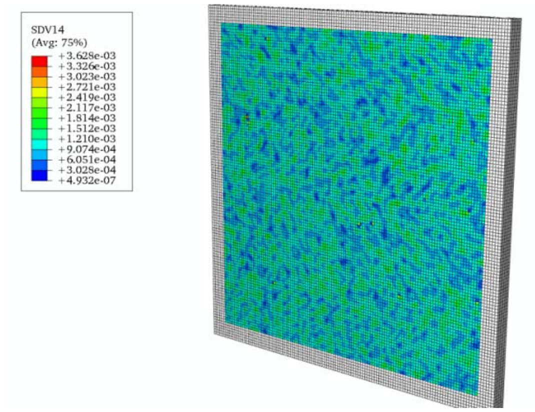

Figure A.4 Calculation 2 initial damage.

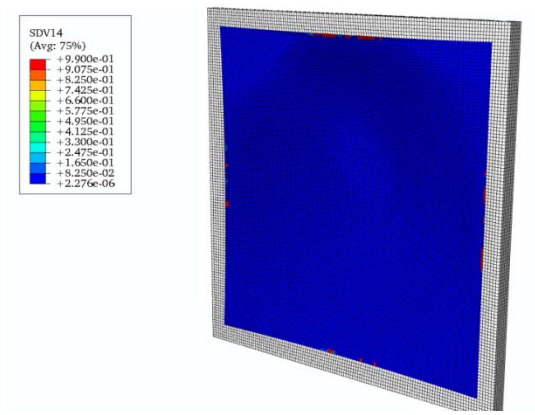

Figure A.5 Calculation 2 final damage front.

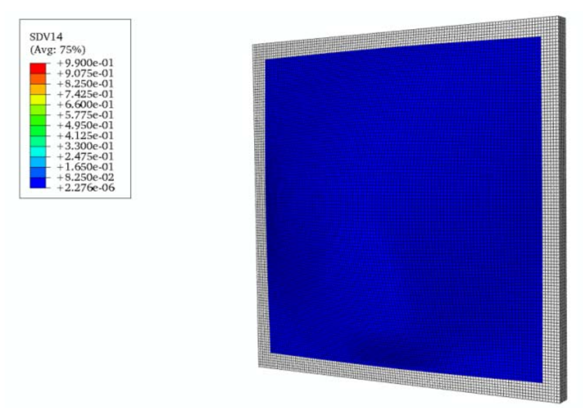

Figure A.6 Calculation 2 final damage back.

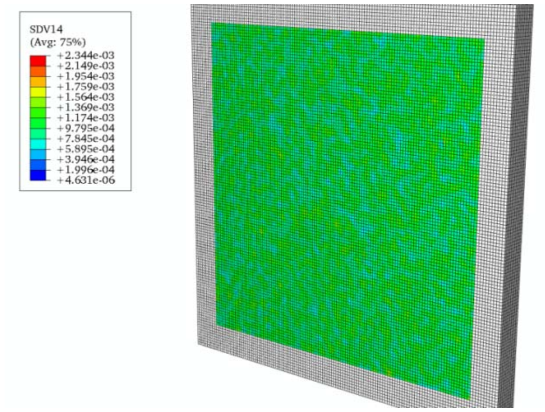

Figure A.7 Calculation 3 initial damage.

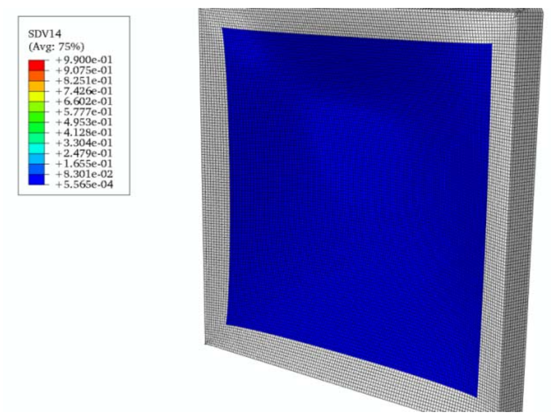

Figure A.8 Calculation 3 final damage front.

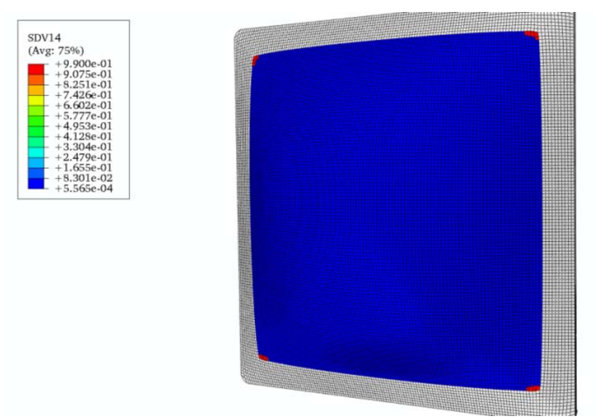

Figure A.9 Calculation 3 final damage back.

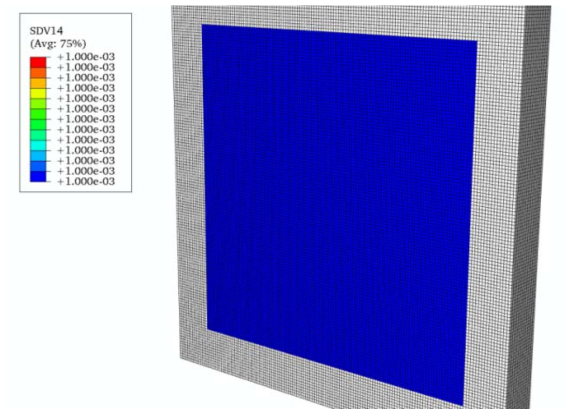

Figure A.10 Calculation 4 initial damage.

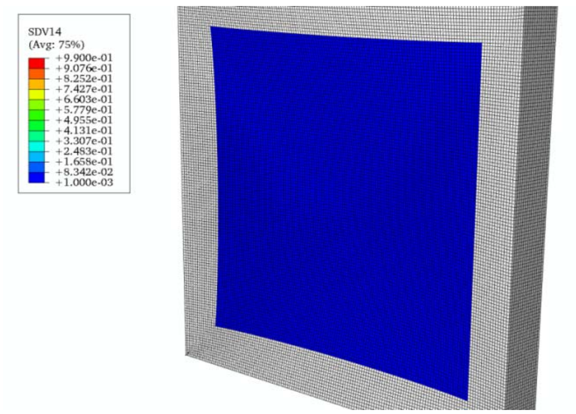

Figure A.11 Calculation 4 final damage front.

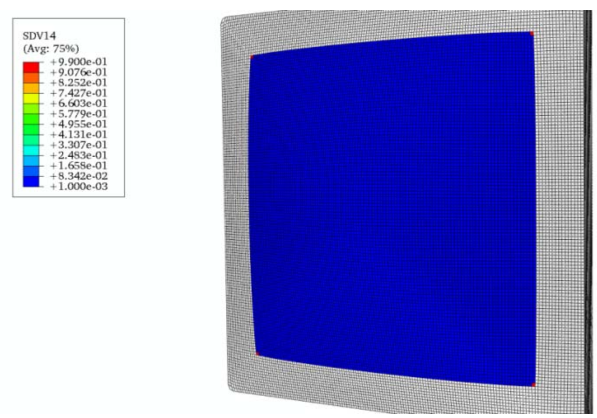

Figure A.12 Calculation 4 final damage back.

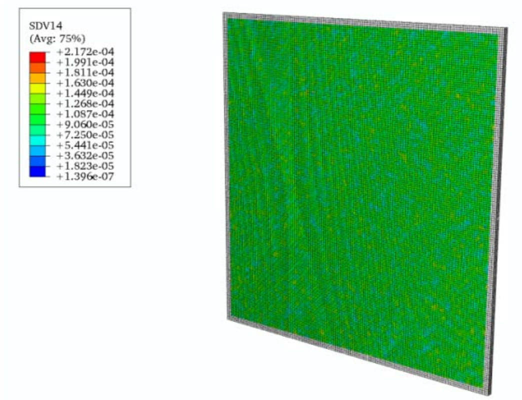

Figure A.13 Calculation 5 initial damage.

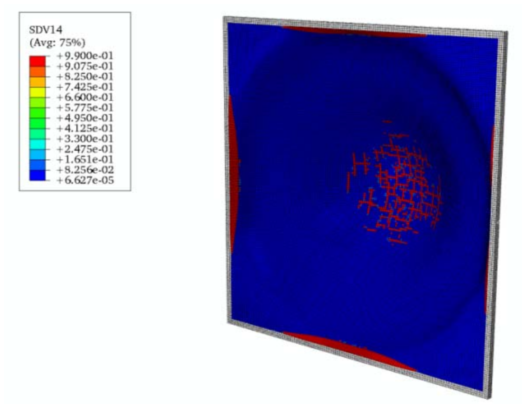

Figure A.14 Calculation 5 final damage front.

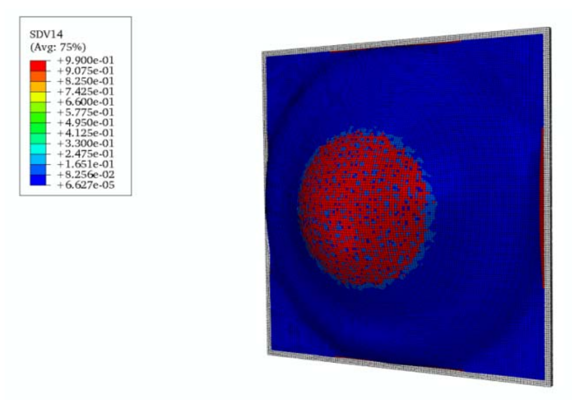

Figure A.15 Calculation 5 final damage back.

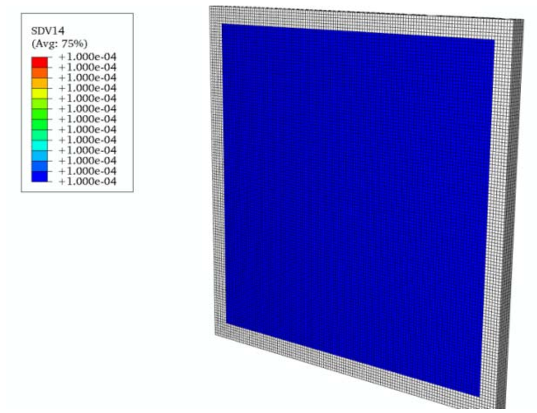

Figure A.16 Calculation 6 initial damage.

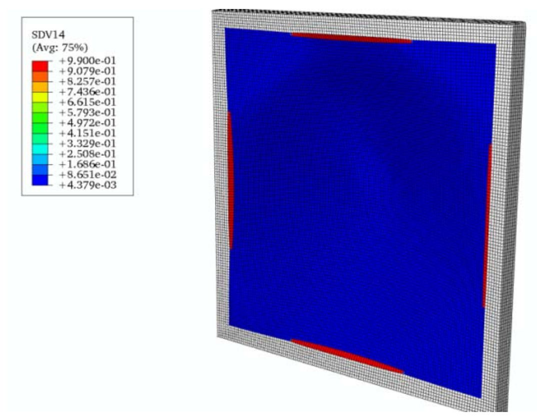

Figure A.17 Calculation 6 final damage front.

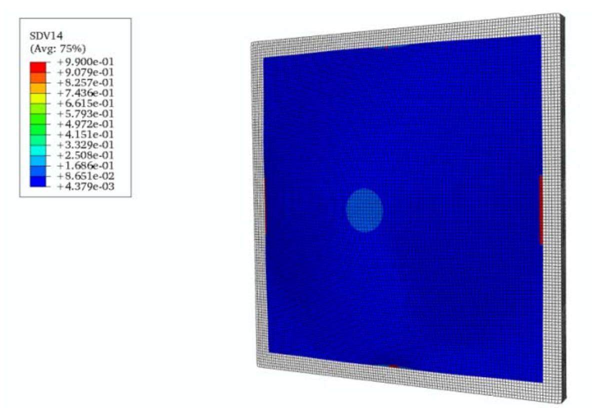

Figure A.18 Calculation 6 final damage back.

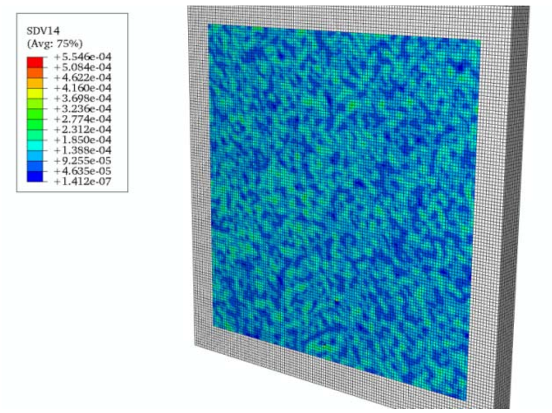

Figure A.19 Calculation 7 initial damage.

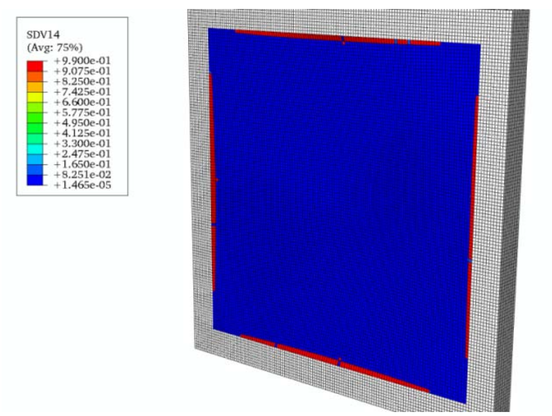

Figure A.20 Calculation 7 final damage front.

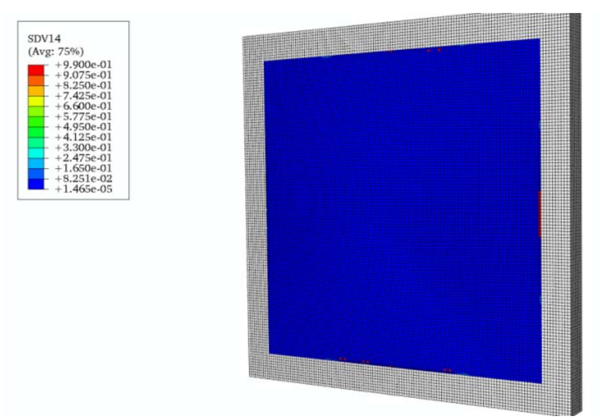

Figure A.21 Calculation 7 final damage back.

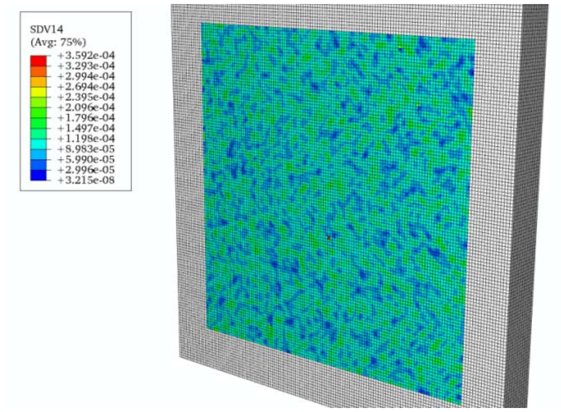

Figure A.22 Calculation 8 initial damage.

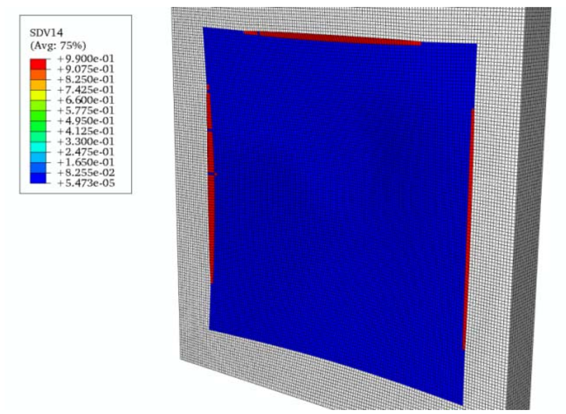

Figure A.23 Calculation 8 final damage front.

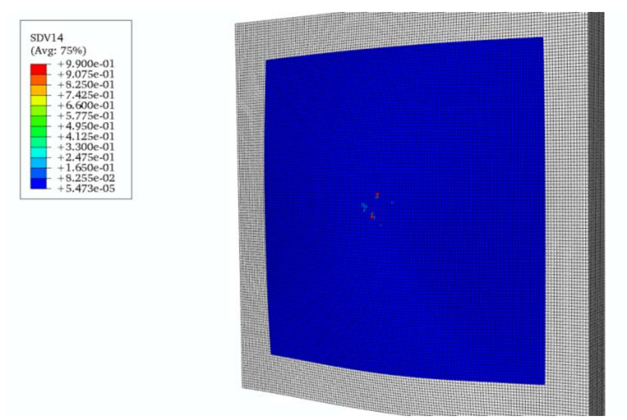

Figure A.24 Calculation 8 final damage back.

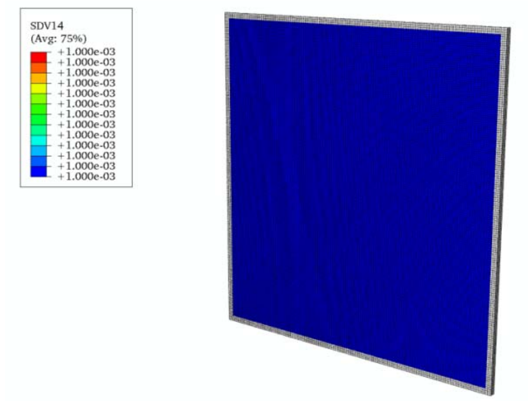

Figure A.25 Calculation 9 initial damage.

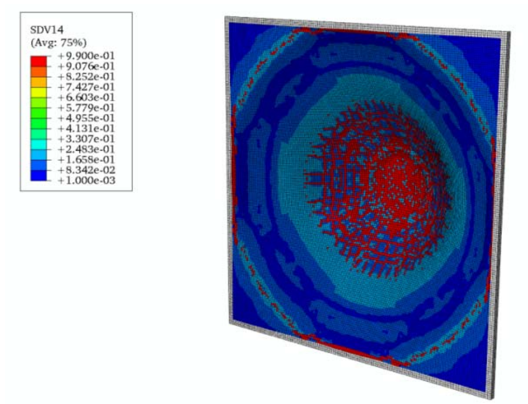

Figure A.26 Calculation 9 final damage front.

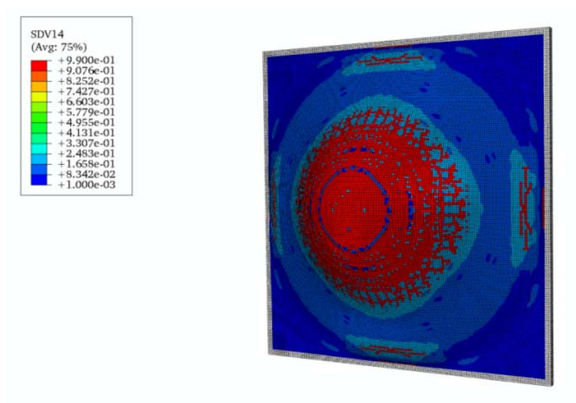

Figure A.27 Calculation 9 final damage back.

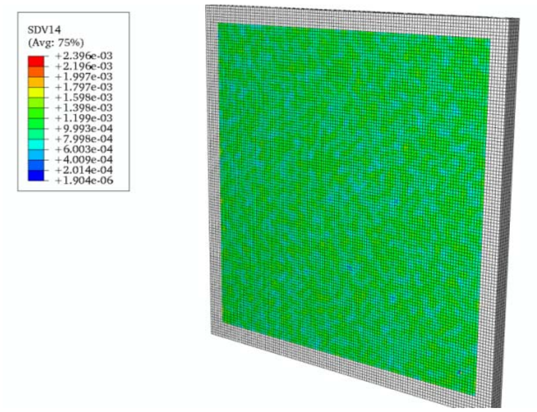

Figure A.28 Calculation 10 initial damage.

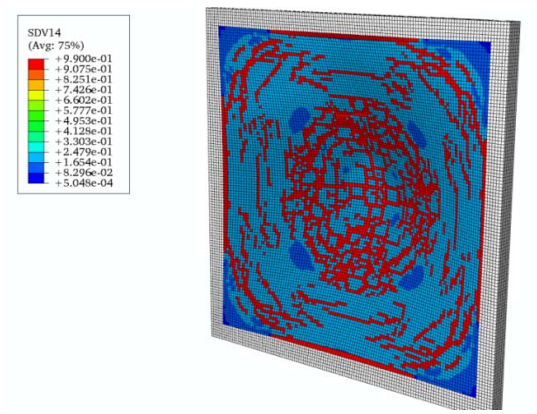

Figure A.29 Calculation 10 final damage front.

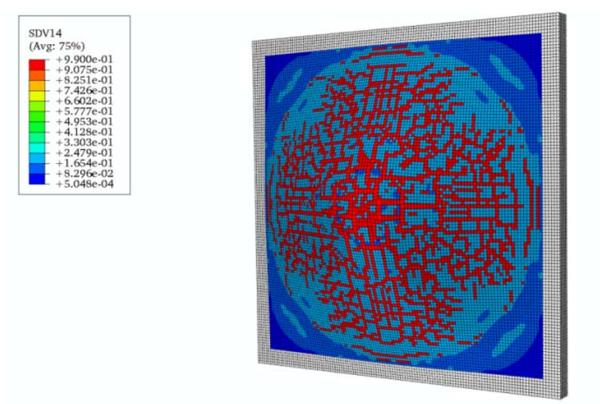

Figure A.30 Calculation 10 final damage back.

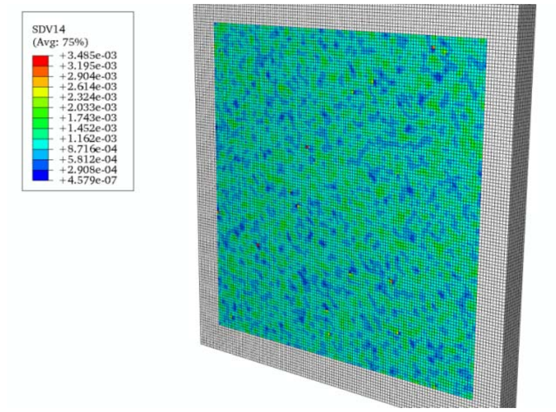

Figure A.31 Calculation 11 initial damage.

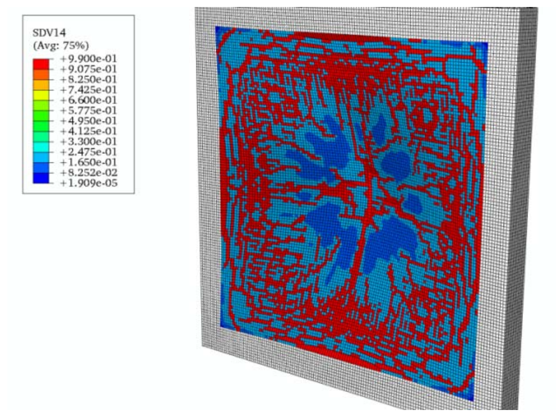

Figure A.32 Calculation 11 final damage front.

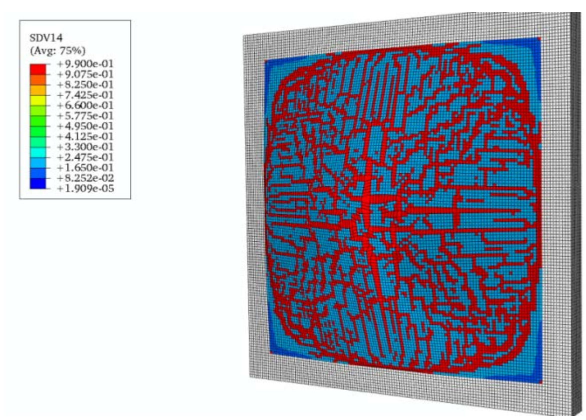

Figure A.33 Calculation 11 final damage back.

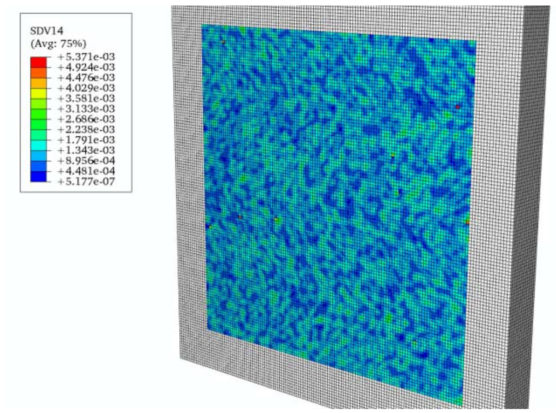

Figure A.34 Calculation 12 initial damage.

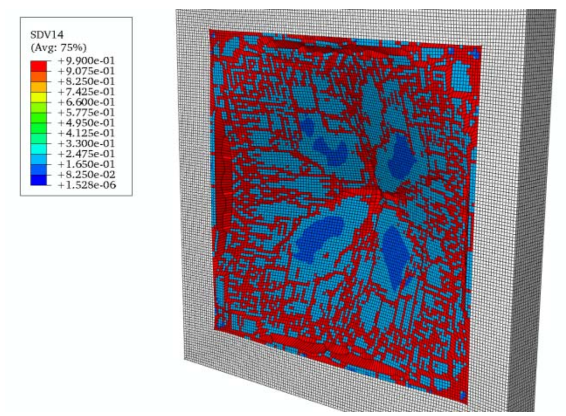

Figure A.35 Calculation 12 final damage front.

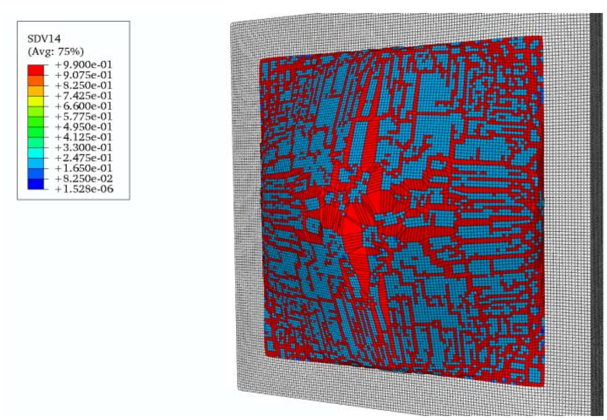

Figure A.36 Calculation 12 final damage back.

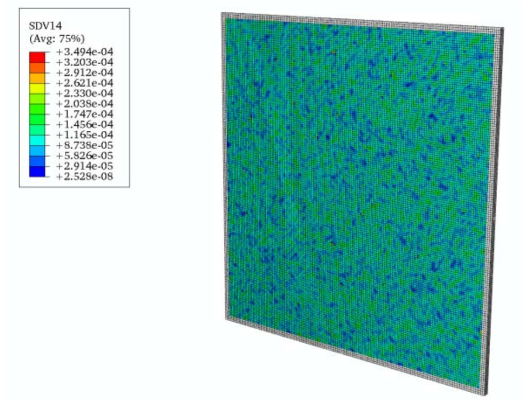

Figure A.37 Calculation 13 initial damage.

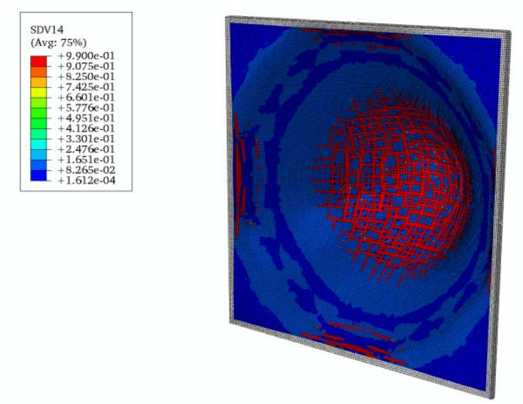

Figure A.38 Calculation 13 final damage front.

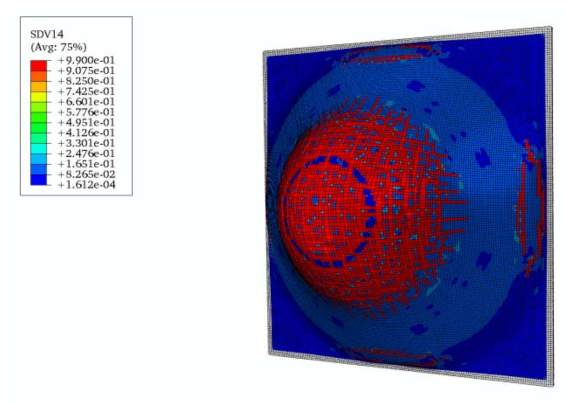

Figure A.39 Calculation 13 final damage back.

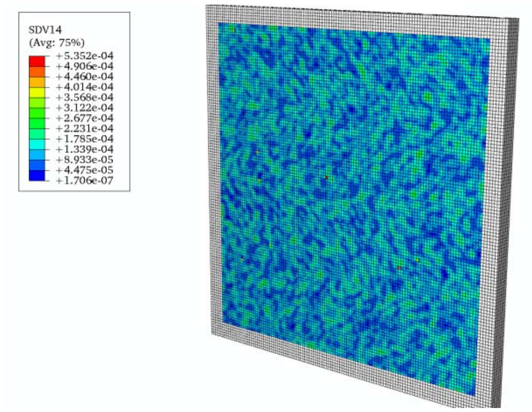

Figure A.40 Calculation 14 initial damage.

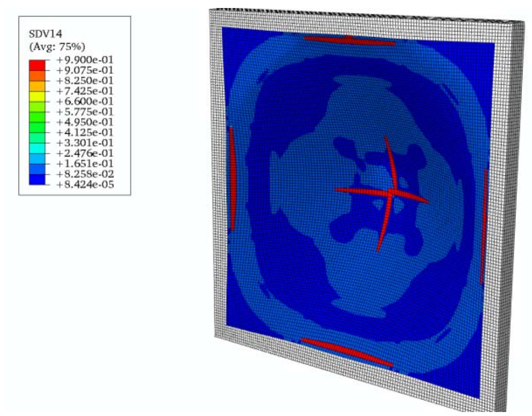

Figure A.41 Calculation 14 final damage front.

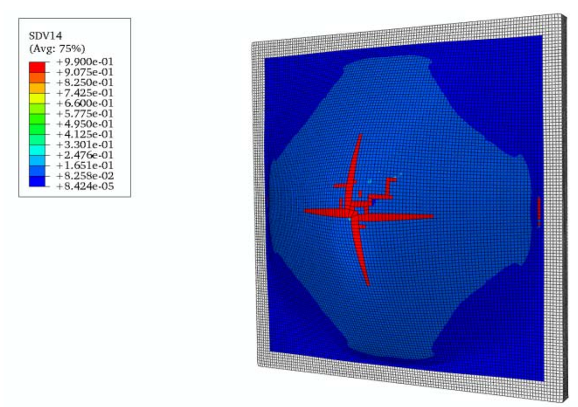

Figure A.42 Calculation 14 final damage back.

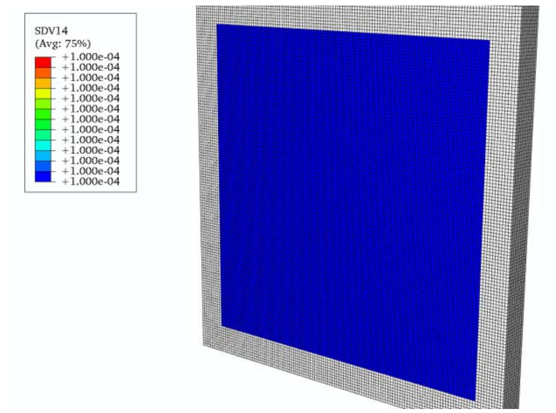

Figure A.43 Calculation 15 initial damage.

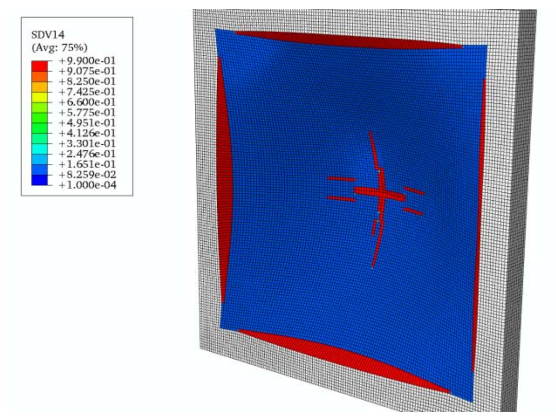

Figure A.44 Calculation 15 final damage front.

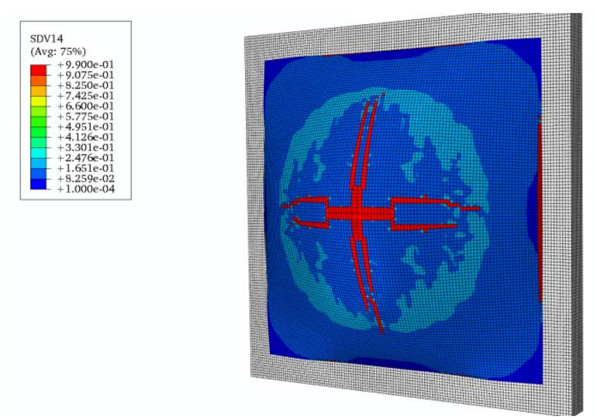

Figure A.45 Calculation 15 final damage back.

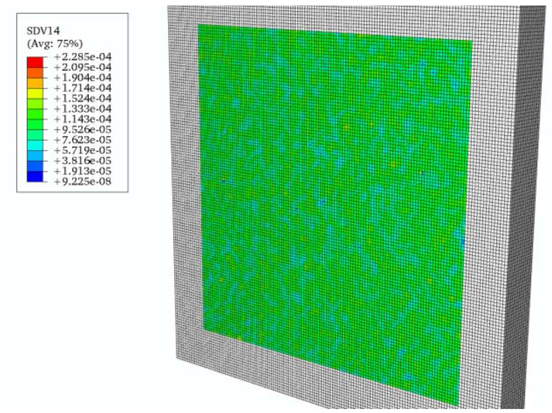

Figure A.46 Calculation 16 initial damage.

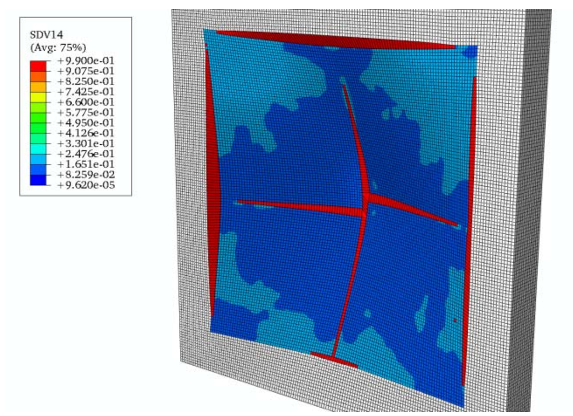

Figure A.47 Calculation 16 final damage front.

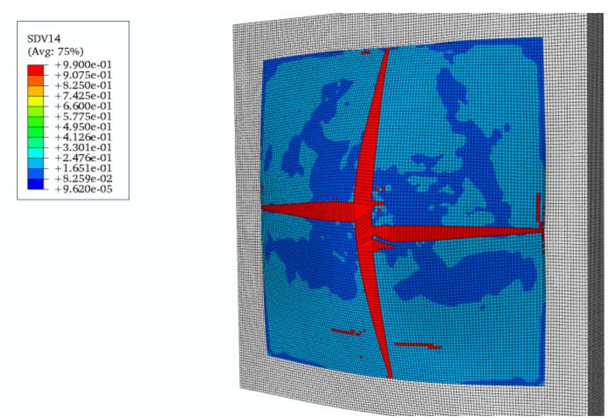

Figure A.48 Calculation 16 final damage back.

APPENDIX B

ABAQUS VDLOAD SUBROUTINE

```
C
C User subroutine VDLOAD
      subroutine vdload (
C Read only (unmodifiable) variables -
     *     nblock, ndim, stepTime, totalTime,
     *     amplitude, curCoords, velocity, dircos,
     *     jltyp, sname,
C Write only (modifiable) variable -
     *     value )
cc
      include 'vaba_param.inc'
cc
      dimension curCoords(nblock,ndim),
     *     velocity(nblock,ndim),
     *     dircos(nblock,ndim,ndim),
     *     value(nblock)
cc
      character*13 shkwv_data
      character*70 :: press_file_dir = 'directory'
      integer, parameter :: max = 5000
      integer :: i, j, k, valnum, filenum
      real, dimension(max) :: xcoord, ycoord, zcoord
      real, dimension(40,max) :: p, t
      real, parameter :: x_ab_cen = 0.0d0, y_ab_cen = 0.0d0
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCC
C     Open shockwave amplitude file and save values
C     to time and pressure arrays
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCC
      do i = 1, 40
        write(shkwv_data, 20) i
        open(10,file=press_file_dir//shkwv_data, status='old', err=30)
        read(10, *)
        read(10, *) xcoord(i)
        read(10, *)
        read(10, *) ycoord(i)
        read(10, *)
        read(10, *) zcoord(i)
        read(10, *)
        j = 1
        do while (.true.)
         read(10, *, end = 10) t(i,j), p(i,j)
         j = j + 1
         valnum = j - 1
         filenum = i
        end do
10      close(10)
      end do
20    format( '15kg_press', i3.3 )
30    close(10)
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCC
C     Apply the blast
C     load to the plate
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCC
      do k = 1, nblock
        x_ab = curCoords(k,1)
        y_ab = curCoords(k,2)
        r_ab = sqrt((x_ab-x_ab_cen)**2+(y_ab-y_ab_cen)**2)
        if (r_ab < xcoord(filenum)) then
          do i = 1, filenum
            if (r_ab >= xcoord(i) .and. r_ab < xcoord(i+1)) then
              do j = 1, valnum
                if (totalTime < t(i, valnum)) then
                  if (totalTime >= t(i,j) .and.
     *              totalTime < t(i,j + 1)) then
                    p1 = p(i,j) + (totalTime - t(i,j)) *
     *                ((p(i,j + 1) - p(i,j))/(t(i, j + 1) - t(i,j)))
                  end if
                else
                  p1 = p(i,valnum)
                end if
              end do
              do j = 1, valnum
                if (totalTime < t(i+1, valnum)) then
                  if (totalTime >= t(i+1,j) .and.
     *              totalTime < t(i+1,j+1)) then
                    p2 = p(i+1,j) + (totalTime - t(i+1,j)) *
     *                ((p(i+1,j + 1)-p(i+1,j))/
     *                (t(i+1,j + 1) - t(i+1,j)))
                    value(k) = p1 + (r_ab - xcoord(i)) *
     *                ((p2 - p1)/(xcoord(i+1) - xcoord(i)))
                    goto 40
                  end if
                else
                  p2 = p(i+1,valnum)
                  value(k) = p1 + (r_ab - xcoord(i)) *
     *              ((p2 - p1)/(xcoord(i+1) - xcoord(i)))
                  goto 40
                end if
              end do
            end if
          end do
        else
          value(k) = 0
        end if
40    end do
      return
      end subroutine vdload
```

APPENDIX C

PYTHON SCRIPT TO CONVERT ABAQUS INPUT FILE TO LS-DYNA INPUT FILE

```
#Copyright Kyle Crosby
#Version 1.0
#7/6/2009

#Converts Abaqus input file to LS-Dyna input file.

import linecache
import math

input_file = raw_input('Enter input file name: ')
node_file = open('node.nd', 'w')
element_file = open('element.el', 'w')
ns_file = open('node_sets.ns', 'w')

#Find line number for Abaqus key words
node_line_number = []
element_line_number = []
ns_line_number = []
instance_line_number = []
for i, info in enumerate(open(input_file, 'r')):
    if "*Part" in info:
        node_line_number.append(i+1)
    if "*Element" in info:
        element_line_number.append(i+1)
    if "*Nset" in info:
        ns_line_number.append(i+1)
    if "*Instance" in info:
        instance_line_number.append(i+1)
print('Keyword Locations Found')

#Find part translation and rotation and save to a list.
translation_data = {}
m = 0
n = instance_line_number[m]

while m < len(instance_line_number):
    try:
        if "*" in linecache.getline(input_file, n + 1):
            zero = [0.0, 0.0, 0.0]
            translation_data[m] = zero
            m = m + 1
            n = instance_line_number[m]
        else:
            raw_translation_data = linecache.getline(input_file, n + 1)
            raw_translation_data = eval(raw_translation_data)
            translation_data[m] = list(raw_translation_data)
            m = m + 1
            n = instance_line_number[m]
    except IndexError:
        break

rotation_data = {}
```

```
m = 0
n = instance_line_number[m]

while m < len(instance_line_number):
    try:
        if "*" in linecache.getline(input_file, n + 2):
            zero = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
            rotation_data[m] = zero
            m = m + 1
            n = instance_line_number[m]
        else:
            raw_rotation_data = linecache.getline(input_file, n + 2)
            raw_rotation_data = eval(raw_rotation_data)
            rotation_data[m] = list(raw_rotation_data)
            m = m + 1
            n = instance_line_number[m]
    except IndexError:
        break

# Rotates and translates nodes to global coordinates, and
# writes node data to node file.
node_data = []
j = 0
i = node_line_number[j]

node_file.write('*Node\n$   node              x              y' + \
               '              z      tc      rc\n')

while j < 10000:
    try:
        while i < 100000:
            node_data = linecache.getline(input_file, i + 2)
            node_data = eval(node_data)
            node_data = list(node_data)

            a = rotation_data[j][0]
            b = rotation_data[j][1]
            c = rotation_data[j][2]
            d = rotation_data[j][3]
            e = rotation_data[j][4]
            f = rotation_data[j][5]

            u = d - a
            v = e - b
            w = f - c
            theta = rotation_data[j][6] * (math.pi / 180)

            x = node_data[1] + translation_data[j][0]
            y = node_data[2] + translation_data[j][1]
            z = node_data[3] + translation_data[j][2]
            L = math.sqrt((u**2) + (v**2) + (w**2))
```

```
            try:
                x_rotated = a * (v**2 + w**2) + u * (-b*v - c*w + u*x + v*y + \
                        w*z) + ((x - a) * (v**2 + w**2) + u * (b*v + c*w - \
                        v*y - w*z)) * math.cos(theta) + L * (b*w - c*v - w*y + \
                        v*z) * math.sin(theta) / L**2

                y_rotated = b * (u**2 + w**2) + v * (-a*u - c*w + u*x + v*y + \
                        w*z) + ((y - b) * (u**2 + w**2) + v * (a*u + c*w - \
                        u*x - w*z)) * math.cos(theta) + L * (-a*w + c*u + w*x - \
                        u*z) * math.sin(theta) / L**2

                z_rotated = c * (u**2 + v**2) + w * (-a*u - b*v + u*x + v*y + \
                        w*z) + ((z - c) * (u**2 + v**2) + w * (a*u + b*v - \
                        u*x - v*y)) * math.cos(theta) + L * (a*v - b*u - v*x + \
                        u*y) * math.sin(theta) / L**2
                node_file.write('%8i%16E%16E%16E       0       0\n' % \
                        ((node_data[0] + (j * 100000)), \
                        x_rotated, y_rotated, z_rotated))
                i = i + 1
            except ZeroDivisionError:
                node_file.write('%8i%16E%16E%16E       0       0\n' % \
                        ((node_data[0] + (j * 100000)), \
                        x, y, z))
                i = i + 1
    except SyntaxError:
        j = j + 1
        try:
            i = node_line_number[j]
        except IndexError:
            break
node_file.write('*End')
print('Node File Written')

# Write element data to element file.
element_data = []
j = 0
i = element_line_number[j]

element_file.write('*ELEMENT_SOLID\n$    eid     pid      n1      n2      n3' + \
             '      n4      n5      n6      n7      n8\n')

while j < 10000:
    try:
        while i < 100000:
            element_data = linecache.getline(input_file, i + 1)
            element_data = eval(element_data)
            element_data = list(element_data)
            element_file.write('%8i%8i%8i%8i%8i%8i%8i%8i%8i%8i\n' % \
                        ((element_data[0] + (j * 100000)), \
                        (j + 1), \
                        (element_data[1] + (j * 100000)), \
                        (element_data[2] + (j * 100000)), \
```

```
                        (element_data[3] + (j * 100000)), \
                        (element_data[4] + (j * 100000)), \
                        (element_data[5] + (j * 100000)), \
                        (element_data[6] + (j * 100000)), \
                        (element_data[7] + (j * 100000)), \
                        (element_data[8] + (j * 100000))))
            i = i + 1
    except SyntaxError:
        j = j + 1
        try:
            i = element_line_number[j]
        except IndexError:
            break
element_file.write('*End')
print('Element File Written')

# Write node set data to node set file.
# This section isn't correct right now
# due to the way the elements are named.
ns_data = []
j = 0
i = ns_line_number[j]

while j < len(ns_line_number):
    try:
            ns_file.write('*Set_Node_List\n' + '       ' + str(j + 5) + \
                    ',       0,' + '       0,' + '       0,' + \
                    '       0\n')
        if "*" in linecache.getline(input_file, i + 1):
            j = j + 1
            i = ns_line_number[j]
        while i < 100000:
            ns_data = linecache.getline(input_file, i + 1)
            ns_data = eval(ns_data)
            ns_data = list(ns_data)
            ns_file.write('%8s,%8s,%8s,%8s,%8s,%8s,%8s,%8s\n\
%8s,%8s,%8s,%8s,%8s,%8s,%8s,%8s\n' % \
                    ((ns_data[0] + (j * 100000)), \
                    (ns_data[1] + (j * 100000)), \
                    (ns_data[2] + (j * 100000)), \
                    (ns_data[3] + (j * 100000)), \
                    (ns_data[4] + (j * 100000)), \
                    (ns_data[5] + (j * 100000)), \
                    (ns_data[6] + (j * 100000)), \
                    (ns_data[7] + (j * 100000)), \
                    (ns_data[8] + (j * 100000)), \
                    (ns_data[9] + (j * 100000)), \
                    (ns_data[10] + (j * 100000)), \
                    (ns_data[11] + (j * 100000)), \
                    (ns_data[12] + (j * 100000)), \
                    (ns_data[13] + (j * 100000)), \
                    (ns_data[14] + (j * 100000)), \
```

```
                (ns_data[15] + (j * 100000))))
            i = i + 1
    except IndexError:
        break
ns_file.write('*End')
print('Node Set File Written')

node_file.close()
element_file.close()
ns_file.close()
```

APPENDIX D

SAMPLE ABAQUS INPUT FILE

```
********************************************************************************
*HEADING
Shock Wave Impacting Plate
**in mm, N, tonne(10^3 kg), s, MPa(N/mm^2), mJ, tonne/mm^3
********************************************************************************
*NODE, INPUT=10mm.nd
*INCLUDE,INPUT=node_sets.ns
*ELEMENT, INPUT=10mm.el, TYPE=C3D8R
*solidsection, elset=plate, controls=EC-1, material=Magnesium
1.,
*INCLUDE,INPUT=magnesium_am60.mtrl
*solidsection, elset=boundary, controls=EC-1, material=Boundary
1.,
*Material, Name=Boundary
*Density
 7.850E-09
*Elastic
 10000, 0.03
*Include, Input=el_sets.es
*sectioncontrols, name=EC-1, Hourglass=Viscous
1.0, 1.0, 1.0
*boundary
Fixed, 1, 3, 0
*Initial Conditions, Type=Solution, INPUT=solution.dat
*surface,type=element,name=plate_surf
surface, S3
********************************************************************************
*Step, name=Step-1
*dynamic, explicit, Direct User Control
1.5E-07,0.00035
*DSLoad
plate_surf, PNU, 1
*output, field, Num=150
*nodeoutput
U, V, A, RF
*elementoutput
S, LE, SDV, ER, PE, ELEN
*Output, History
*Energy Output, Variable=Preselect, Elset=plate
*endstep
```

APPENDIX E

PYTHON SCRIPT TO CREATE ABAQUS SOLUTION FILE

```
#Imports Abaqus node and element files and outputs Abaqus solution file.
#Copyright Kyle Crosby ("ha")
#Version 3.0
#This version is for use with 25 dependant variables and can read element
#labels from element sets.
#1/15/09

import linecache
import random
import math


#graph = open('graph.dat', 'w')
solution_file = open('solution.dat', 'w')
input_file_type = raw_input('Enter number of file type:\n1. Element file' + \
                    '\n2. Element Set File\n')

if input_file_type == "1":
    node_input = raw_input('Enter node file name: ')
    element_input = raw_input('Enter element file name: ')
    temp = raw_input('Enter initial temperature: ')
    distribution_type = raw_input('Enter number of damage distribution type: ' + \
                    '\n' + '1. Uniform Distribution' + '\n' + \
                    '2. Random Normal Distribution' + "\n")
elif input_file_type == "2":
    es_file = raw_input('Enter element set file name: ')
    temp = raw_input('Enter initial temperature: ')
    distribution_type = raw_input('Enter number of damage distribution type: ' + \
                    '\n' + '1. Uniform Distribution' + '\n' + \
                    '2. Random Normal Distribution' + "\n")
    es_line_number = []
    for i, info in enumerate(open(es_file, 'r')):
        if '*Elset, elset=plate' in info:
            es_line_number.append(i+1)
else:
    print('Invalid value for file type')

if distribution_type == "1":
    uniform_value = raw_input('Enter value for uniform damage: ')
elif distribution_type == "2":
    normal_mean = float(input('Enter mean value for normally distributed ' + \
                'damage: '))
    normal_stdev = float(input('Enter standard deviation for normally ' + \
                'distributed damage: '))
else:
    print('Invalid value for distribution type')

if input_file_type == "1":
#Import node file
    node_data = {}
    i = 1
    while i < 2000000:
```

```
        node_file = linecache.getline(node_input, i)
        if node_file == "":
            break
        node_file = eval(node_file)
        node_file = list(node_file)
        node_index = node_file[0]
        node_coordinates = node_file[1:4]
        node_data[node_index] = node_coordinates
        i = i + 1
#Checks the number of lines that one line of element data takes up.
#This affects the interval used in the loop.
        line_check = linecache.getline(element_input, 2)
        line_check = eval(line_check)
    try:
        line_check = list(line_check)
        num = 1
    except TypeError:
        num = 2
#Calculate element centroid location
    j = 1
    while j < 2000000:
        element_data = linecache.getline(element_input, j)
        if element_data == "":
            break
        element_data = eval(element_data)
        element_data = list(element_data)
        del(element_data[2:5])
        element_name = element_data[0]
#Find x coordinate of Node 1
        node_1_x = node_data[element_data[1]][0]
#Find x, y, and z coordinate of Node 5
        node_5_x = node_data[element_data[2]][0]
        node_5_y = node_data[element_data[2]][1]
        node_5_z = node_data[element_data[2]][2]
#Find y coordinate of Node 6
        node_6_y = node_data[element_data[3]][1]
#Find z coordinate of Node 7
        node_7_z = node_data[element_data[4]][2]
        x_centroid = (node_1_x + node_5_x)/2
        y_centroid = (node_5_y + node_6_y)/2
        z_centroid = (node_5_z + node_7_z)/2
#Print solution and graph file
#Solution file is file used in Abaqus with the *Solution keyword
#Graph file is file with x and y coordinte of element centroid and
#element damage value used for graphical representation of damage
        if distribution_type == "1":
            solution_file.write(str(element_name) + ",,,,,,,\n " + str(temp) + \
                                ",,,,,," + str(uniform_value) + ",\n ,," + \
                                str(uniform_value) + ",,,,,\n ,\n")
            element_data = str(element_data[:]).strip('[]')
#         graph.write(str(x_centroid) + ",   " + \
#                 str(y_centroid) + ",   " + \
```

```
#               str(uniform_value) + "\n")
        if distribution_type == "2":
            rand_number = str(abs(random.normalvariate(normal_mean,normal_stdev)))
            solution_file.write(str(element_name) + ",,,,,,,\n " + str(temp) + ",,,,,," + \
                                rand_number + ",\n ,," + rand_number + ",,,,,\n ,\n")
            element_data = str(element_data[:]).strip('[]')
#          graph.write(str(x_centroid) + ",  " + \
#               str(y_centroid) + ",  " + \
#               str(random.normalvariate(0.0001,0.0001)) + "\n")
        print(element_name)
        del(element_data)
        j = j + num

elif input_file_type == "2":
    element_name = []
    i = es_line_number[0] + 1
    while i < 100000:
        try:
            element_name = linecache.getline(es_file, i)
            if '*' in linecache.getline(es_file, i):
                break
            else:
                element_name = eval(element_name)
                element_name = list(element_name)
                j = 0
                while j < 8:
                    if distribution_type == "1":
                        solution_file.write(str(element_name[j]) + ",,,,,,,\n " + str(temp) + \
                                            ",,,,,," + str(uniform_value) + ",\n ,," + \
                                            str(uniform_value) + ",,,,,\n ,\n")
                    if distribution_type == "2":
                        rand_number = str(abs(random.normalvariate(normal_mean,normal_stdev)))
                        solution_file.write(str(element_name[j]) + ",,,,,,,\n " + str(temp) + ",,,,,," + \
                                            rand_number + ",\n ,," + rand_number + ",,,,,\n ,\n")
                    print(element_name[j])
                    j = j + 1
                i = i + 1
        except SyntaxError:
            break
#graph.close()
solution_file.close()
```

APPENDIX F

DMG MODEL MATERIAL CONSTANTS

Table F.1 DMG model (elastic-plastic) constants for AM60 Mg Alloy.

AM60 Mg	Constants	Value
Constants for J/B formulas for G and K	G (MPa)	12810
	a	1
	Bulk (MPa)	38440
	b	0
	melt temp (K)	5556
Specifies the yield stress	C1 (MPa)	2.66
	C2 (K)	0
	C3 (MPa)	92.82
	C4 (K)	47.93
	C5 (1/MPa)	0.00001
	C6 (K)	6.991E-07
Kinematic hardening and recovery terms	C7 (1/MPa)	1.929E+07
	C8 (K)	6868
	C9 (MPa)	1577
	C10 (K)	0.6931
	C11 (sec/MPa)	6.529E-05
	C12 (K)	1.064E+06
Isotropic hardening and recovery terms	C13 (1/MPa)	14.8
	C14 (K)	6.911E-07
	C15 (MPa)	40770
	C16 (K)	102.4
	C17 (sec/MPa)	0
	C18 (K)	0
Yield strength adjustment terms	C19	0
	C20	0
Hardening and recovery cons.	Ca	1.883
	Cb	0.008272
Temperature	init. Temp (K)	297
	heat gen. Coeff.	0.34

Table F.2 DMG model (damage) constants for AM60 Mg Alloy.

AM60 Mg	Constants	Value
McClintock void growth	Void Growth exp	0.246
	Init. rad. (mm)	0.0002
Nucleation	a	1
	b	1
	c	1
	Nuc coeff	0
	Fract. Toughness MPa (m^1/2)	17.3
	Part. Size (mm)	0.0004
	Part. vol fract.	0.07
Coalescence	cd1	0.7
	cd2	1
	dcs0 (mm)	20
	dcs (mm)	20
	dcs exp zz	0.0509
CA pore growth	Init. Void vol fract.	0.001
Nucleation	Nuc temp dependence	0
Coalescence	Coal temp dependence	0

Table F.3 DMG model (elastic-plastic) constants for AZ31 Mg Alloy.

AZ31 Mg	Constants	Value
Constants for J/B formulas for G and K	G (MPa)	16667
	a	0
	Bulk (MPa)	50000
	b	0
	melt temp (K)	5556
Specifies the yield stress	C1 (MPa)	4.8
	C2 (K)	0
	C3 (MPa)	125
	C4 (K)	0
	C5 (1/MPa)	1
	C6 (K)	0
Kinematic hardening and recovery terms	C7 (1/MPa)	1
	C8 (K)	0
	C9 (MPa)	1950
	C10 (K)	0
	C11 (sec/MPa)	0
	C12 (K)	0
Isotropic hardening and recovery terms	C13 (1/MPa)	0.08
	C14 (K)	0
	C15 (MPa)	1200
	C16 (K)	0
	C17 (sec/MPa)	0
	C18 (K)	0
Yield strength adjustment terms	C19	0
	C20	3
Hardening and recovery cons.	Ca	0
	Cb	0.9
Temperature	init. Temp (K)	297
	heat gen. Coeff.	0.34

Table F.4 DMG model (damage) constants for AZ31 Mg Alloy.

AZ31 Mg	Constants	Value
McClintock void growth	Void Growth exp	0.3
	Init. rad. (mm)	0.003
Nucleation	a	0
	b	20
	c	10
	Nuc coeff	4.2
	Fract. Toughness MPa (m^1/2)	24.9
	Part. Size (mm)	0.0056
	Part. vol fract.	0.0085
Coalescence	cd1	20
	cd2	0
	dcs0 (mm)	0.003
	dcs (mm)	0.003
	dcs exp zz	0
CA pore growth	Init. Void vol fract.	0.0001
Nucleation	Nuc temp dependence	0
Coalescence	Coal temp dependence	0

Table F.5 DMG model (elastic-plastic) constants for A356 Al Alloy.

A356 T6	Constants	Value
Constants for J/B formulas for G and K	G (MPa)	27200
	a	0
	Bulk (MPa)	70890
	b	0
	melt temp (K)	1089
Specifies the yield stress	C1 (MPa)	1.5
	C2 (K)	0
	C3 (MPa)	255
	C4 (K)	0
	C5 (1/MPa)	1
	C6 (K)	0.000E+00
Kinematic hardening and recovery terms	C7 (1/MPa)	1.400E+00
	C8 (K)	0
	C9 (MPa)	3160
	C10 (K)	0
	C11 (sec/MPa)	0.000E+00
	C12 (K)	0.000E+00
Isotropic hardening and recovery terms	C13 (1/MPa)	0.2
	C14 (K)	0.000E+00
	C15 (MPa)	2300
	C16 (K)	0
	C17 (sec/MPa)	0
	C18 (K)	0
Yield strength adjustment terms	C19	0
	C20	0
Hardening and recovery cons.	Ca	0
	Cb	-0.2
Temperature	init. Temp (K)	297
	heat gen. Coeff.	0.39

Table F.6   DMG model (damage) constants for A356 Al Alloy.

A356 T6	Constants	Value
McClintock void growth	Void Growth exp	0.4
	Init. rad. (mm)	0.003
Nucleation	a	530
	b	450
	c	184
	Nuc coeff	10
	Fract. Toughness MPa (m^1/2)	17.3
	Part. Size (mm)	0.006
	Part. vol fract.	0.07
Coalescence	cd1	0.85
	cd2	0
	dcs0 (mm)	0.03
	dcs (mm)	0.03
	dcs exp zz	0
CA pore growth	Init. Void vol fract.	0.001
Nucleation	Nuc temp dependence	0
Coalescence	Coal temp dependence	0

Table F.7 DMG model (elastic-plastic) constants for 6061 Al Alloy.

6061 T651	Constants	Value
Constants for J/B formulas for G and K	G (MPa)	25900
	a	0
	Bulk (MPa)	67500
	b	0
	melt temp (K)	855
Specifies the yield stress	C1 (MPa)	2.2
	C2 (K)	0
	C3 (MPa)	255
	C4 (K)	0
	C5 (1/MPa)	1
	C6 (K)	0
Kinematic hardening and recovery terms	C7 (1/MPa)	2.2
	C8 (K)	0
	C9 (MPa)	10000
	C10 (K)	0
	C11 (sec/MPa)	0
	C12 (K)	0
Isotropic hardening and recovery terms	C13 (1/MPa)	0.2
	C14 (K)	0
	C15 (MPa)	1500
	C16 (K)	0
	C17 (sec/MPa)	0
	C18 (K)	0
Yield strength adjustment terms	C19	0
	C20	0
Hardening and recovery cons.	Ca	0
	Cb	-0.4
Temperature	init. Temp (K)	297
	heat gen. Coeff.	0.372

Table F.8 DMG model (damage) constants for 6061 Al Alloy.

6061 T651	Constants	Value
McClintock void growth	Void Growth exp	0.3
	Init. rad. (mm)	0.0013
Nucleation	a	185
	b	130
	c	60
	Nuc coeff	7.25
	Fract. Toughness MPa (m^1/2)	29
	Part. Size (mm)	0.0013
	Part. vol fract.	0.00085
Coalescence	cd1	1.4
	cd2	0
	dcs0 (mm)	0.003
	dcs (mm)	0.003
	dcs exp zz	0
CA pore growth	Init. Void vol fract.	0.0001
Nucleation	Nuc temp dependence	0
Coalescence	Coal temp dependence	0

APPENDIX G

DMG PYTHON POINT SIMULATOR

```
#
#    Python point simulator for DMG damage model
#    Written by Kyle Crosby
#

from linecache import *
from math import *
from pylab import *

input_file = raw_input('Enter parameter file name: ')

num_sim = eval(raw_input('Enter Number of Simulations to be run: '))
total_epslon = eval(raw_input('Enter Final Strain for Simulations: '))
nstep = eval(raw_input('Enter Number of Steps for Simulations: '))

#  Read number of parameters
num_param = getline(input_file, 2)
num_param = eval(num_param)

i = 1
props = []
props.append(0.0)

while i <= num_param:
   j = 2*(i+1)
   props_data = getline(input_file, j)
   if input_file == "":
      break
   props_data = eval(props_data)
   props.append(props_data)
   i = i + 1

sim_run = 1
while sim_run <= num_sim:

#   rate = eval(raw_input('Enter Strain Rate for this Simulation: '))
   if sim_run==1:
      rate = 0.001
   if sim_run==2:
      rate = -0.001
   if sim_run==3:
      rate = 1000
   if sim_run==4:
      rate = -3900

   if rate >= 0.0:
      test_type = 'Tension'
   else:
      test_type = 'Compression'

   output_file = open('output'+str(sim_run)+'.csv', 'w')
   output_file.write('Test Type\n'+test_type+'\n')
```

```
output_file.write('Strain Rate\n'+str(rate)+'\n')
output_file.write('True Strain,True Stress,Total Damage,Nucleation\n')

nucleation = []
plot_strain = []

ttotal  = total_epslon/rate       #total time
dt      = ttotal/nstep            #time step
eng_str = rate*dt                 #engineering strain

cc1   = props[6]
cc2   = props[7]
cc3   = props[8]
cc4   = props[9]
cc5   = props[10]
cc6   = props[11]
cc7   = props[12]
cc8   = props[13]
cc9   = props[14]
cc10  = props[15]
cc11  = props[16]
cc12  = props[17]
cc13  = props[18]
cc14  = props[19]
cc15  = props[20]
cc16  = props[21]
cc17  = props[22]
cc18  = props[23]
cc19  = props[24]
cc20  = props[25]
ca    = props[26]
cb    = props[27]
tempi = props[28]
htcp  = props[29]
cd1   = props[39]
cd2   = props[40]
dcs0  = props[41]
dcs   = props[42]
zz    = props[43]

vmcc10 = pi*(props[31]**2)       # McClintock void growth(second phase pores)
vmcc11 = 0.0                     # rate of change of M porosity
vnuc13 = props[35]               # nucleation
dam14 = props[35]*vmcc10         # damage
vnuc17 = 0.0                     # nucleation from previous time step
coash18 = props[44]              # Cocks-Ashby void growth(large pores)
coash19 = 0.0

theta = tempi
if (tempi == 0.):
   theta = temp
```

```
    if (props[5] == 0.0):
        blk = props[3]
        g = props[1]
    else:
        tratio = theta/props[5]
        tratio = min(tratio,0.9999)
        g = props[1]*(1.0-tratio*exp(props[2]*(1.0-1.0/tratio)))

    twog = 2.0 * g
    blk = props[3] - props[4]*tratio
#    Young's Modulus
    young = (9.0*g*blk)/(g+3.0*blk)
    iplast=0
    sig   = 0.0            # Stress
    atr   = 0.0            # Alpha
    xktr  = 0.0            # Kappa
    epto  = 0.0            # Total Strain
    time  = 0.0            # total step time
    eps   = 0.0
    epst  = 0.0
    drate = 0.0            # Damage rate:state(16)
    tot_eng_str = 0.0

    icycle = 1

    while icycle <= nstep:       # Iteration Cycle

        tot_true_str_old = log(1+tot_eng_str)
        tot_eng_str = tot_eng_str + eng_str
        tot_true_str_new = log(1+tot_eng_str)    #true strain
        true_str_inc = tot_true_str_new - tot_true_str_old
        epslon = true_str_inc                    #strain increments

        time = time+dt
        sig1 = sig
        epto = tot_eng_str
#---- damage
        dam1 = 1.0-dam14
#---- check for melt
#

        if(theta < props[5]):
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#            Yield and Rate-Dependant Yield Terms
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
# Loading Parameter (+ for tension, - for compression)
            tsion = (rate/abs(rate))*(2.0/(3.0*sqrt(3.0)))
            ztheta = 1.0/theta
# Anisotropic Hardening
#           ytheta = cc3*exp(cc4*ztheta)
```

```
#          vtheta = cc1*exp(-cc2*ztheta)
#          ftheta = cc5*exp(-cc6/theta)
#          rd1 = cc7*(1.0-cb*tsion)*exp(-cc8/theta)
#          h1 = (cc9-cc10*theta)*(1.0+cb*tsion)
#          rs1 = cc11*exp(-cc12*ztheta)
#          rd2 = cc13*(1.0-cb*tsion)*exp(-cc14/theta)
#          h2 = (cc15-cc16*theta)*(1.0+cb*tsion)
#          rs2 = cc17*exp(-cc18*ztheta)
#         h3 = 0.0
#Anisotropic Yielding
        ytheta = cc3*exp(cc4*ztheta)*(1.0+cb*tsion)
        vtheta = cc1*exp(-cc2*ztheta)*(1.0+cc20*tsion)
        ftheta = cc5*exp(-cc6*ztheta)
        rd1 = cc7*exp(-cc8/theta)
        h1 = (cc9-cc10*theta)
        rs1 = cc11*exp(-cc12*ztheta)
        rd2 = cc13*exp(-cc14*ztheta)
        h2 = (cc15-cc16*theta)
        rs2 = cc17*exp(-cc18*ztheta)
        h3 = 0.0
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++

        ddd = epslon/dt

#          den = (dcs0/dcs)**zz
        den = 1.0

        alpm = sqrt(2.0/3.0)*abs(atr)
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                         trial alpha
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        atr1=atr
        atr = atr*(1.0-(dt*(rs1+(rd1*ddd))*abs(atr)*den))
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                         trial kappa
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        xktr1 = xktr
        xktr = xktr*(1.0-(dt*(rs2+(rd2*ddd))*xktr*den))
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                         yield radius
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        if (ftheta == 0.0):
           belog = 1.0
        else:
           belog = log((ddd+sqrt(ddd**2+ftheta**2))/ftheta)
```

```
      ak = ((vtheta*belog)+ytheta+xktr)*dam1
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                    trial elastic stress
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#      dam2 = 1.0-min(1.0,dt*drate/dam1)
      dam2 = 1.0-(dt*drate/dam1)
#
      sig = (dam2*sig)+(dam1*young*epslon)
#
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
      ximag11 = sig-atr

      ximag2 = ximag11**2

      ak2 = ximag2-(ak*abs(ak))


      if (ak2 <= 0.0) and (iplast == 0):
        output_file.write(str(epto)+','+str(sig)+','+str(dam14)+
                         ','+str(vnuc13)+'\n')
        continue
      else:
        iplast=1
        ximag = sqrt(ximag2)
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                       dgam
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        dter = den*(h1+h2*dam1)
        dte1 = (dam1*young)+dter
        dgam = (ximag-ak)/dte1
        dgam2 = dgam/ximag
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        dsig = dam1*young*dgam2
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                      stress
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        sig = sig-(dsig*ximag11)
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                      kappa
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        xktr = xktr+(dgam*h2*den)
```

```
        xktr = max(0.0,xktr)
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        dalph = (h1+h3)*dgam2
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                 alpha
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        atr = atr + h1*dgam2*ximag11*den
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                 eps
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        eps = eps+dgam
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#        Update temperature for adiabatic problems
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        theta = theta+(htcp*dgam2*sig*ximag11)
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                epsdot
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        epsdot= dgam/dt
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                cacon
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        cacon = abs(vtheta/ytheta)
        if (cacon <= 0.0):
          cacon=11.0
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                beta
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
        rrat = (1.0/3.0)*sqrt(2.0/3.0)
        dterm=2.0*(2.0*cacon-1.0)/(2.0*cacon+1.0)
        arg = min(15.0,dterm*rrat)
        beta = sinh(max(0.,arg) )
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#                Cocks-Ashby
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#    Cocks-Ashby large pore growth term
        phi1 = 1.0-coash18
```

```
            c90 = 1.0 + cacon
            psi = min(15.0,beta*dt*epsdot*c90)
            tmp = max(0.0,(1.0+(phi1**c90-1.0)*exp(psi)))
            coash18 = min((1.0-tmp**(1.0/c90)),.99)
#     Cocks-Ashby void growth rate
            coash19 = beta*epsdot*(1.0/(1.0-coash18)**(vtheta/ytheta)
                   -(1.0-coash18))
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#     McClintock form of void growth
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
            abc = (3.0**0.5/(2.0*(1.0-props[30]))*sinh(3.0**0.5*
                 0.5*(1.0-props[30])*((2.0*rrat)+(1.0/3.0))))
            vrad   = props[31]*exp(eps*abc/sqrt(2.0/3.0))
            vmcc10 = pi*(vrad**2)
            vmcc11 = 3.0*vmcc10*abc*epsdot
            p30 = 1.0-props[30]
            ddt1 = -sqrt(3.0)/(sqrt(2.0/3.0)*p30)

#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#     nucleation
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
            tn1   = (props[33]*tsion)+props[34]
            tn1 = abs(tn1)
            zzzz  = (props[37]**0.5/(props[36]*props[38]**(1.0/3.0)))*tn1
            vnuc17 = vnuc13
            j2 = (1.0/3.0)*sig**2.0
            vnuc13 = (props[35]*exp(abs(vtheta/ytheta)*j2
                              *eps*zzzz/sqrt(2.0/3.0))
                              *exp(-props[45]/theta))
            nucleation.append(vnuc13)
#     added for nonmonotonic path sequences, statev(17) is old nucleation
            if(vnuc13 < vnuc17):
               vnuc13 = abs(vnuc17+vnuc13)
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++
#     Coalescence factor
            cf=(cd1+cd2*vnuc13*vmcc10)*exp(props[46]*theta)*den
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++
#     Damage
            damage = cf*(vnuc13*vmcc10+coash18)
            if(damage >= 0.20):
               damage = .99
            dam14 = min(damage,0.99)

            if(dam14 >= 0.99):
               drate = 0.0
               sig = 0.0
```

```
                output_file.write(str(epto)+','+str(sig)+','+str(dam14)+
                           ','+str(vnuc13)+'\n')
                break
             else:
#     Nucleation Rate
                epsdot=abs(epsdot)
                rnuc15=zzzz*vnuc13*epsdot
#     Damage Rate
                zsecond=cf*((rnuc15*vmcc10)+(vnuc13*vmcc11)+coash19)
                zthird=(((vnuc13*vmcc10)+coash18)*
                       cd2*((dcs0/dcs)**zz)*exp(0.0*theta)*
                       ((rnuc15*vmcc10)+(vnuc13*vmcc11)))
                drate = zsecond+zthird
                output_file.write(str(epto)+','+str(sig)+','+str(dam14)+
                              ','+str(vnuc13)+'\n')

       else:
          atr  = 0.0
          xktr = 0.0
          sig  = 0.0
          eps  = 0.0

       icycle = icycle + 1

   output_file.close()
   sim_run = sim_run + 1

#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#   Plots Nucleation Data
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
i = 1
while i <= num_sim:
   mod_data = 'output'+str(i)+'.csv'
   test_type = getline(mod_data, 2).rstrip()
   strain_rate = eval(getline(mod_data, 4))
   figure(1)
   nucl_model = plotfile(mod_data, ('true_strain', 'nucleation'),newfig=False, lw=2,
                   label=test_type+' Nucleation Model for '+str(abs(strain_rate))
                   +' /s', skiprows=4)
   xlabel('Strain')
   ylabel('Nucleation')
   ylim(0, 80)
   xlim(0, total_epslon)
   title('Nucleation')
   legend(loc='lower right')
   grid(True)
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#   Plots Model Stress-Strain Data
```

```
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
    figure(2)
    model_line = plotfile(mod_data, ('true_strain', 'true_stress'), lw=2,
                          label=test_type+' Model for '+str(abs(strain_rate))+' /s',
                          newfig=False, skiprows=4)
    i = i + 1
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++
#   Plots Experimental Stress-Strain Data
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++

num_exp_data = raw_input('Enter Number of Experimental Data\n'
                         'files to be entered: ')

i = 1
num_exp_data = eval(num_exp_data)
while i <= num_exp_data:
    exp_data = raw_input('Enter file name: ')
    mat_name = getline(exp_data, 2).rstrip()
    test_type = getline(exp_data, 4).rstrip()
    strain_rate = eval(getline(exp_data, 6))
    data_line = plotfile(exp_data, ('strain', 'stress'),
                 newfig=False, skiprows=6, lw=2,
                 label=test_type+' Data for '+str(abs(strain_rate))+' /s')
    i = i + 1

xlabel('Strain')
ylim(0, 500.0)
xlim(0, total_epslon)
ylabel('Stress (MPa)')
title('Stress-Strain Curve')
legend(loc='lower right')
grid(True)

show()
```